HAL
open science

Efficient Multiscale Sauvola's Binarization

Guillaume Lazzara, Thierry Géraud

To cite this version:

Guillaume Lazzara, Thierry Géraud. Efficient Multiscale Sauvola's Binarization. International Journal on Document Analysis and Recognition, 2014, 17 (2), pp.105-123. 10.1007/s10032-013-0209-0 . hal-02181880

HAL Id: hal-02181880

https://hal.science/hal-02181880

Submitted on 12 Jul 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

International Journal on Document Analysis and Recognition manuscript No.
(will be inserted by the editor)

Efficient Multiscale Sauvola's Binarization

Guillaume Lazzara · Thierry Géraud

Received: date / Accepted: date

Abstract This work focuses on the most commonly used binarization method: Sauvola's. It performs relatively well on classical documents, however, three main defects remain: the window parameter of Sauvola's formula does not fit automatically to the contents, it is not robust to low contrasts, and it is not invariant with respect to contrast inversion. Thus on documents such as magazines, the contents may not be retrieved correctly, which is crucial for indexing purpose.

In this paper we describe how to implement an efficient multiscale implementation of Sauvola's algorithm in order to guarantee good binarization for both small and large objects inside a single document without adjusting manually the window size to the contents. We also describe how to implement it in an efficient way, step by step. This algorithm remains notably fast compared to the original one.

For fixed parameters, text recognition rates and binarization quality are equal or better than other methods on text with low and medium x-height and is significantly improved on text with large x-height. Pixel-based accuracy and OCR evaluations are performed on more than 120 documents. Compared to awarded methods in the latest binarization contests, Sauvola's formula does not give the best results on historical documents. On the other hand, on clean magazines it outperforms those methods.

This implementation improves the robustness of Sauvola's algorithm by making the results almost insensible to the window size whatever the object sizes. Its properties make it usable in full document analysis toolchains.

Keywords Binarization, Multiscale, Document Image Analysis, Algorithm

Guillaume Lazzara
E-mail: guillaume.lazzara@lrde.epita.fr

Thierry Géraud
E-mail: thierry.geraud@lrde.epita.fr

EPITA Research and Development Laboratory (LRDE)
14-16, rue Voltaire, FR-94276 Le Kremlin-Bicêtre, France
Tel: +33 1 53 14 59 22, Fax: +33 1 53 14 59 13

1 Introduction

1.1 Overview

Over the last decades, the need for document image analysis has increased significantly. One critical step of the analysis is to identify and retrieve foreground and background objects correctly. One way to do it is to produce a binary image; however it is not easy to find the best thresholds because of change of illumination or noise presumed issues. As exposed in Sezgin and Sankur's survey [1], many attempts have been made to find an efficient and relevant binarization method.

Some methods performs globally. Otsu's algorithm [2] is known as one of the best in that category. It aims at finding an optimal threshold for the whole document by maximizing the separation between two pre-assumed classes. Despite fast computing times, it is not well adapted to uneven illumination and to the presence of random noise.

Other methods performs locally, trying to find the different satisfying thresholds for specific regions or around every pixels. A well-performing local thresholding method was proposed by Niblack [3]. The idea is to compute an equation usually based on the mean and the standard deviation of a small neighborhood around each pixel. It works fine on clean documents but can give deceiving results in relatively degraded documents. Then,

Sauvola and Pietikainen [4] proposed an improvement of Niblack's method to improve binarization robustness on noisy documents or when show-through artifacts are present. Currently this is one of the best binarization methods for classical documents according to several surveys [1, 5]. While Niblack's and Sauvola's methods rely on the local variance, other local methods use different local features like the contrast in Bernsen's method [6]. Some methods also try to mix global and local approaches, like Gabarra's [7], so that object edges are detected and the image is split into regions thanks to a quadtree. Depending on whether the region contains an edge or not, a different threshold formula is used.

Local methods are more robust than global methods but often introduce parameters. Usually, methods are parameter-sensitive and the most difficult part is to find the best values for the set of documents to be processed. Algorithms for automatic estimation of free parameter values have been proposed by Badekas and Papamarkos [5] and Rangoni et al. [8]. Unfortunately, even if these values fit many kinds of documents, they may not be generic enough and some adaptations may be needed with respect to the type of documents to process. That is the main reason why many attempts have been made to improve the best methods by automatically adjusting the parameters to the global [9, 10] or local contents [11, 12]. This also includes some works on getting multiscale versions of common algorithms like Otsu's or Sauvola's [10]. Eventually, improvements can be effectively observed in specific cases.

Over the last four years, more and more local methods tries to rely not only on the pixel values in the threshold decision but also on higher-level information. Lu et al. [13] models the document background via an iterative polynomial smoothing and then choose local thresholds based on detected text stroke edges. Lelore and Bouchara [14, 15] use coarse thresholding to partition pixels into three groups: ink, background, and unknown. Some models describe the ink and background clusters, and guide decisions on the unknown pixels. Because they rely on the document contents, those methods are usually considered as parameters free. Furthermore, the recent contests have proven their efficiency on historical documents: Lu's method won DIBCO 2009[16] and an improved version tied as a winner of HDIBCO 2010[17], whereas Lelore's method won DIBCO 2011[18]. More recently, the winner of HDIBCO 2012, Howe, proposes a method [19] which optimizes a global energy function based on the Laplacian image. It uses both a Laplacian operator to assess the local likelihood of foreground and background labels and Canny edge detection to identify likely discontinuities. Finally, a graph cut implementation finds the minimum energy solution of a function combining these concepts. Parameters of the method are also adjusted dynamically w.r.t the contents using a stability criterion on the final result.

Because Sauvola's binarization is widely used in practice and gives good results on magazines, this paper focuses on that particular method.

1.2 Sauvola's Algorithm and Issues

Sauvola's method [4] takes a grayscale image as input. Since most of document images are color images, converting color to grayscale images is required [10]. For this purpose, we choose to use the classical luminance formula, based on the eye perception:

$$\text{Luma} = 0.299 \times \text{R} + 0.587 \times \text{G} + 0.114 \times \text{B}.$$

From the grayscale image, Sauvola proposed to compute a threshold at each pixel using:

$$T = m \times \left[1 + k \times \left(\frac{s}{R} - 1\right)\right]. \tag{1}$$

This formula relies on the assumption that text pixels have values close to black (respectively background pixels have values close to white). In Equation 1 k is a user-defined parameter, m and s are respectively the mean and the local standard deviation computed in a window of size w centered on the current pixel and R is the dynamic range of standard deviation ($R = 128$ with 8-bit gray level images). The size of the window used to compute m and s remains user-defined in the original paper.

Combined with optimizations like integral images [20], one of the main advantages of Sauvola's method is its computational efficiency. It can run in less than 60 ms on A4 300-dpi documents with a modern computer. Another advantage is that it performs relatively well on noisy and blurred documents [1].

Due to the binarization formula, the user must provide two parameters (w, k). Some techniques have been proposed to estimate them. Badekas and Papamarkos [5] state that $w = 14$ and $k = 0.34$ is the best compromise for show-through removal and object retrieval quality in classical documents. Rangoni et al. [8] based the parameter research on Optical Character Recognition (OCR) result quality and found $w = 60$ and $k = 0.4$. Sezgin and Sankur [1] and Sauvola and Pietikainen [4] used $w = 15$ and $k = 0.5$. Adjusting those free parameters usually requires an *a priori* knowledge on the set of documents to get the best results. Therefore there is no consensus in the research community regarding those parameter values.

Sauvola's method suffers from different limitations among the following ones.

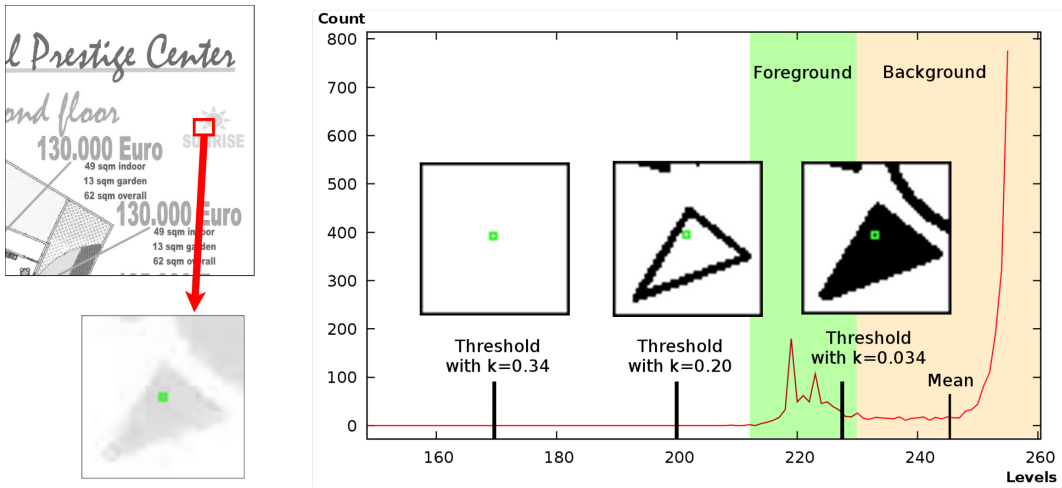


Fig. 1: Influence of the parameter k on the threshold in case of low contrasts. A window of size 51x51 pixels centered on the central point in green is used and the corresponding histogram is computed. k must be very low to extract correctly this pixel inside an object with low contrast.

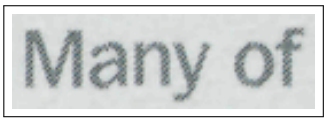
Missing low-contrast objects. Low contrasted objects may be considered as textured background or show-through artifacts due to the threshold formula (Equation 1) and may be removed or partially retrieved. Figure 1 illustrates this issue. The region of interest considered shows the values taken into account in a window of size $w = 51$ centered at the central point depicted in green: contrasts are very low. In that case, the corresponding histogram illustrates how sensitive Sauvola's method is to k. Object pixels cannot be correctly retrieved if k is greater than 0.034. A low value of this parameter can help retrieving low-contrasted objects but since it is set for the whole document, it also alters other parts of the result: correctly contrasted objects are thicker in that case, possibly causing unintended connections between components. This is due to the fact that background noise and artifacts are usually poorly contrasted and are retrieved as objects.

keeping textured text as is. Textures are really sensitive to window size. 2a and 2d show binarization results of textured and non-textured text with the same font size. Even though the textured text is bold, inner parts of the characters are missing after binarization (see 2b). In 2e, the text is still well preserved and suitable for OCR. In 2c, using a larger window may improve the binarization results on textured text. However, this solution cannot be applied if it is mixed with plain text since, as shown in 2f, the retrieved text would be bolded.

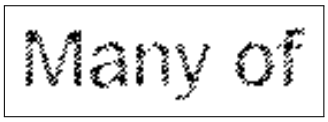
Handling badly various object sizes. In case of both small and large objects in a same document, Sauvola's method will not be able to retrieve all objects correctly. In most cases, one may want to retrieve text in documents, so a small window may be used. Small text should be retrieved perfectly, however larger text may not. 2h illustrates what happens when the selected window is too small compared to the objects of the document. We expect the algorithm to retrieve plain objects but in case of a too small window, statistics inside the objects may behave like in background: pixels values are locally identical. Since Sauvola's formula relies on the fact there is a minimum of contrast in the window to set a pixel as foreground, it is unable to make a proper choice.

Spatial object interference. This issue mainly appears with image captions such as in 3a. Too large windows may include data from objects of different nature. In 3a, data from the image located above the caption is taken into account, leading to irrelevant statistics and invalid binarization. This is probably one of the reasons why Sauvola and Pietikainen [4] chooses to first identify text and non-text regions before binarization.

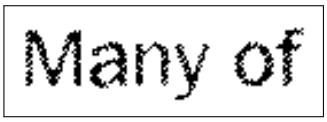
Several attempts have been made in order to improve Sauvola's binarization results and to prevent these defects. Wolf and Jolion [21] try to handle low-contrast and textured text defects. It consists in normalizing the contrast and the mean gray level of the image in order to maximize local contrast. Text is slightly bold though. Bukhari et al. [12] try to improve results by adjusting the parameter k depending on whether a pixel is part of a foreground or background object. They claim that Sauvola's method is very sensible to k and can perform better if it is tuned, which is something we have also



(a) Original grayscale image with textured text.

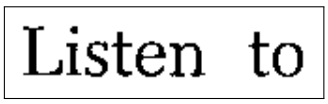


(b) Sauvola's binarization applied on (a) with $w = 11$.



(c) Sauvola's binarization applied on (a) with $w = 51$.

(d) Original grayscale image with small text.



(e) Sauvola's binarization applied on (d) with $w = 11$.

(f) Sauvola's binarization applied on (d) with $w = 51$.

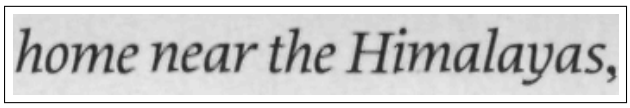
(g) Original grayscale image with large text.

(h) Sauvola's binarization applied on (g) with $w = 51$.

(i) Sauvola's binarization applied on (g) with $w = 501$.

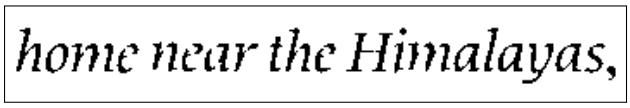
Fig. 2: Influence of Sauvola's algorithm parameters on the results. The size of the window is an important parameter to get good results, too low a value may lead to broken characters and/or characters with holes whereas too large a value may lead to bold characters. Its size must depend on the contents of the document.

(a) Original grayscale image in its context (2215 × 2198 pixels).



(b) Region of interest (351 × 46 pixels).

(c) Sauvola's binarization with $w = 51$.



(d) Sauvola's binarization with $w = 501$.

Fig. 3: Influence of too large a window and object interference. The large picture above the caption introduces a bias in the statistics used to compute the threshold with Sauvola's formula. Taking too much pixels of that picture into consideration can lead to broken or too thin characters.

noticed. Farrahi Moghaddam and Cheriet [10] tried to improve the results in case of intensity and interfering degradation by implementing a multiscale version of Sauvola's algorithm. First, the average stroke width and line height are evaluated. Then, in each step, the scale is reduced by a factor of 2 and the parameters are adjusted: k is set from 0.5 to 0.01. The idea is to make the results from the lower scale grow while retrieving only text pixels at each step. Yet this method only works well on uniform text size.

Kim [22] describes in details issues caused by too small or too large windows. He actually describes some of the limitations cited above and proposes an hybrid solution that takes advantage of two window sizes: a small one in order to get local fine details and a larger one to get the global trend. First, the input image is binarized with a moderate-size window. Then, text lines

are located and features are computed from the text: average character thickness and text height. For each text line, two windows are deduced from those features and two thresholds T_{large} and T_{small} are computed thanks to Sauvola's formula. Finally, the binarization of each text line is performed using:

$$T(x, y) = \alpha T_{large}(x, y) + (1 - \alpha)T_{small}(x, y).$$

According to the author, this method gives better results than Sauvola's binarization. However it introduces a new parameter α which tends to make the fine tuning of the method more difficult, even if the authors claim that the method is not very sensitive to it. Moreover, the critical part of the algorithm remains the text line detection which assumes that the first binarization has retrieved all the text parts and that text components are correctly grouped. In the case of magazines with different kinds of non-text, we have observed that some text components can be grouped with non-text components which may lead to incorrect features and binarization.

In the remainder of this paper, we present an algorithm to overcome one of the four limitations of Sauvola's binarization mentioned previously, e.g., handling various object sizes on a single run of the algorithm, without any prior knowledge on the location of text lines. It is actually penalizing while processing magazines or commercials where text uses different font sizes: titles, subtitles, subscripts, etc. We also focus on the implementation and computational speed which is also a critical aspect from our point of view.

In section 2 we first expose the general principle of the proposed multiscale algorithm. In section 3, we describe implementation details and explain how to implement our method efficiently. section 4 we present some results and compare them to other methods. We conclude on the achievements of this work and discuss future work in section 7.

2 Multiscale binarization

Large text in documents like magazines is of prime importance for indexing purpose since it usually contains the main topics of documents. Among the four presented defects, handling object of different sizes is thus a priority. The problem with binarizing objects of different sizes is caused by using a single window size, thus not well-suited to all objects. A *local* window is needed in order to fit appropriately the local document contents. Since we want to preserve performance, we need to avoid costly pre-processing algorithms which would require additional passes on data. Therefore, we want to include the window selection inside the binarization process, which is possible with a multiscale approach.

Multiscale strategies are common in image processing literature and are sometimes used in binarization. Farrahi Moghaddam and Cheriet [10] starts by processing the full size image with very strict parameters. After each iteration, the input image is subsampled and the parameters are relaxed so that more pixels are considered as foreground. Only those connecting to previously identified components are kept increasing the area of the components found in the previous iterations. Here both data and parameters vary. Tabbone and Wendling propose a method [9] where the image size does not change. A parameter varies in a range of values and the best parameter value is selected by evaluating the result stability at each step. In Gabarra and Tabbone's method [7], edges are detected then a quad-tree decomposition of the input image is computed. On each area, a local threshold is computed and applied to all the pixels of that area. It is multiscale since parts of an image can be processed at different levels in its quad-tree representation.

In our approach, we choose to run the same process at different scales using the same parameters; the input data is just subsampled. Eventually the final result is deduced as a merge of the results obtained at different scales. In our approach, we make the assumption that objects are more likely to be well retrieved at one of the scales. The method is described in details in the following subsections.

2.1 Notation

We will use the following notation:

- uppercase letters refer to image names: S, I, ...
- lowercase letters refer to scalar values: s, n, ...
- subscript values refer to a scale number: I_s is an image I at scale s.
- $I_s(p)$ corresponds to the value of the pixel p of the image I_s.

2.2 General description

The main goal of the proposed method is to find the best scale for which the algorithm is able to decide correctly whether a pixel belongs to the foreground or to the background. As described in subsection 2.4, a relationship exists between the scale where an object is retrieved and the window size which should be used for capturing this object correctly. Our algorithm is composed of four main steps described below and illustrated in Figure 4:

Step 1 *Subsampling.* The input image is successively subsampled to different scales.

Step 2 *Object selection at each scale.* Subsample images are binarized, labeled, and, object components are selected.

Step 3 *Results merging.* Selected objects are merged into a single scale image. A threshold image is deduced from the scale image.

Step 4 *Final binarization.* From the threshold image, the input image is binarized.

2.3 Step 1: Subsampling

First, the input image, once turned into grayscale, I_1 is subsampled at three different scales thus producing three other images: I_2, I_3, and I_4. The choice of the number of scales, here 4, is related to the fact that we work mainly on A4 documents between 300 and 600 dpi. Thus, the maximum size of an object is constrained and, because of the object area range accepted (see subsection 2.4), four scales are sufficient to retrieve correctly objects. However, on larger documents and/or with higher resolutions, it might be useful to have a few more scales.

The reduction factor between scales s and $s+1$ is mostly set to 2. This value has been chosen because higher values may lead to a loss of precision in the final results. This side effect mainly appears for high scales, where images contains less and less information. Using a reduction factor of 3 for the first subsampling is usually fine, if the image has a minimum resolution of 300 dpi. This reduction factor value may also be useful in the case of large documents in order to improve overall performance. At the end of this step, we have four grayscale images: I_1, I_2, I_3 and I_4.

2.4 Step 2: Object Selection at Each Scale

Each input image I_s is processed separately thanks to the same processing chain as depicted in Figure 4. The goal is to select objects within a specific size (area) range.

I_s is binarized using Sauvola's algorithm with a fixed window of size w. As shown in 2h, the size of the window influences the size and the shape of the retrieved objects. Here, the image I_s is a subsampled version of the input and so are the objects. Therefore, working at a scale s with a window of size w is equivalent to work at scale 1 with a window of size $w_{1,s}$ regarding the reduction factor q:

$$w_{1,s} = q^{(s-1)} \times w.$$

When the scale increases, objects size decreases in the subsampled image and objects are more likely to fit the window to avoid the defects shown in 2g.

As shown in Figure 4, the binarization at scale s produces two images: T_s, a threshold image storing the point-wise thresholds used during the binarization; and B_s, the resulting binary image at scale s. T_s will be used later, during the final step. The binarization of I_s includes connected components of various sizes. Some of them need to be removed because they are too small or too large for giving good results with the current window size. We consider a minimum and a maximum size for acceptable objects. We chose the area, i.e. the number of pixels of a connected component, as size criterion. The different ranges of area are defined according to the current scale s, the reduction factor q, and the constant window size w:

- at scale 1:
  $min_area(1) = 0$
  $max_area(1) = w^2 \times 0.7$
- at scale s:
  $min_area(s) = 0.9 \times max_area(s-1)/q^2$
  $max_area(s) = max_area(s-1) \times q^2$
- at the last scale $s_{\max}$:
  $min_area(s_{\max}) = 0.9 \times max_area(s_{\max}-1)/q^2$
  $max_area(s_{\max}) = +\infty.$

Those area ranges correspond to commonly used body text, small titles, and large titles in 300 dpi magazines documents. In order to disambiguate objects which are at the limit between two scales, object area ranges overlap when considered at scale 1.

A connected component labeling is performed on B_s and a selection of components having their area in the expected range is performed. The result is stored in the binary image S_s. At the end of this step, eight images are kept for the next steps: T_1, T_2, T_3, and T_4 store the thresholds; S_1, S_2, S_3, and S_4 store the selection of objects for their corresponding scale.

2.5 Step 3: Result Merging

The main goal of this step (Figure 4) is to prepare the final binarization by mapping each pixel from the input image to a threshold previously computed during Step 2.

Once an object is stored in S_s, it means that it has been retrieved at scale s. One wants to merge this piece of information into a single scalar image E_1. It consists in marking in E_1 each object in S_s using its corresponding scale s (see Figure 5). Since S_1, S_2, S_3, and S_4 are at different scale, objects extracted from S_s images must be rescaled before being marked in E_1.

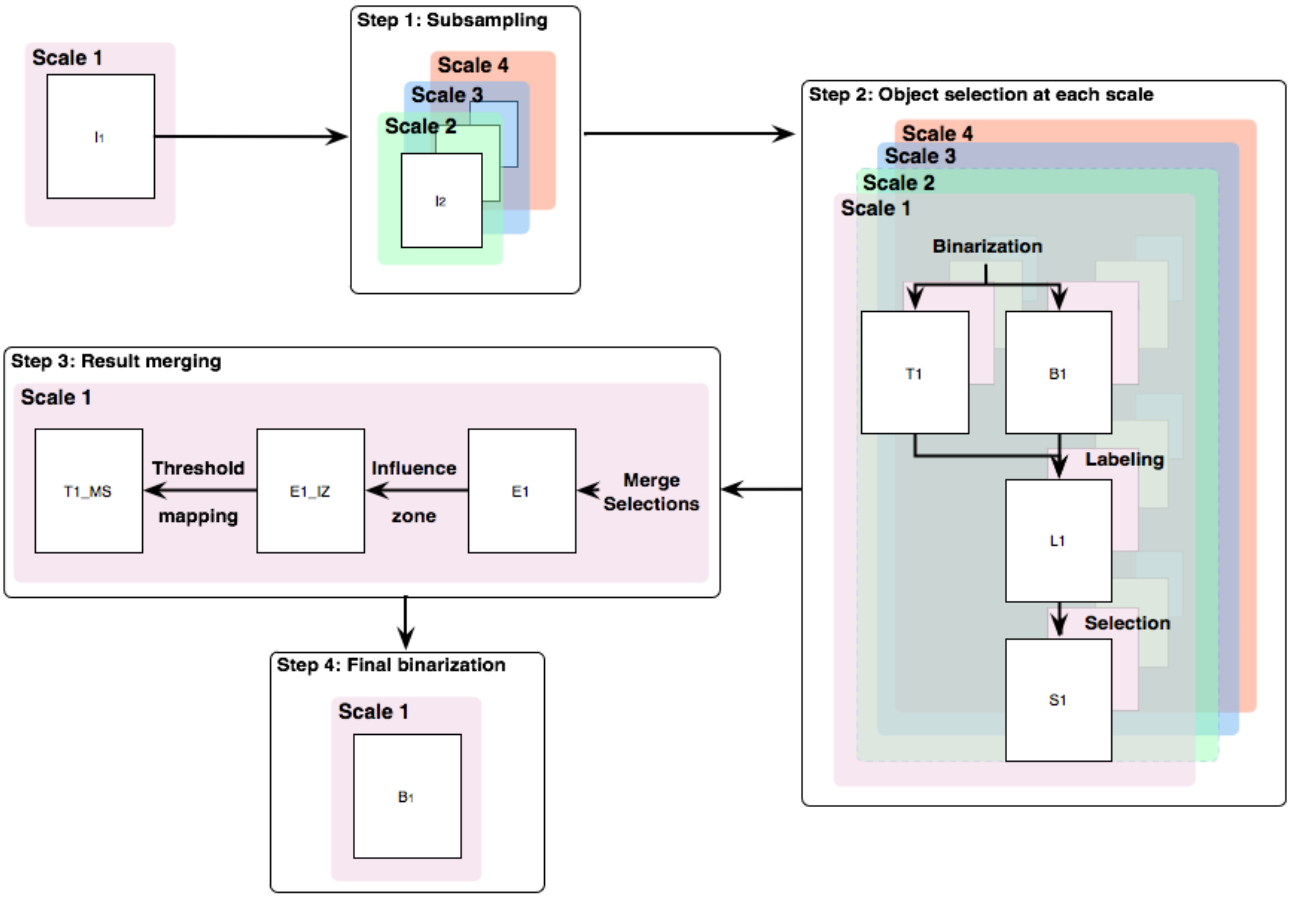


Fig. 4: Conceptual scheme of the method. In Step 1 the input grayscale image is subsampled. In Step 2 each subsampled image is binarized with Sauvola's algorithm and a threshold image T_s is computed and kept for further processing. Objects matching the required area range are considered well-identified and mapped to the current scale. In Step 3 each pixel in the input image is assigned to a scale with respect to the previous results in E_1_IZ. For each pixel p in E_1_IZ, the threshold is deduced by reading the corresponding threshold image T_s where $s = E_1_IZ$ (p). In Step 4 the point-wise binarization is performed using T_1_MS and the input image.

Sometimes objects are retrieved several times at different scales: during subsampling, components may be connected and may have formed objects large enough to enter several area ranges. Overlapping area criteria may also be responsible for such an issue sometimes. In that case, objects are considered to have been retrieved at the highest scale. This way, we guarantee that the window, even if it is not the best one, will be large enough to avoid degraded objects as depicted in 2h.

Once E_1 is computed, every pixels of binarized objects are mapped to a scale. Yet, non-object pixels do not belong to a scale at this step. Most of them are usually background pixels but others can be pixels around objects, ignored because of the loss of precision due to subsampling. For that reason, they must be associated to a scale too in order to be processed like other pixels afterwards. Omitting scale mapping for that pixels and considering them as background information directly would lead to sharp object edges and artifacts.

An influence zone algorithm [23, 24] is applied to E_1 to propagate scale information and guaranty smooth results. It actually consists in a discrete Voronoï tesselation where the seeds are the different connected components. The result is stored in E_1_IZ and values of E_1_IZ are restricted to scale numbers. Here the possible values in that image are 1, 2, 3, and 4. E_1_IZ maps pixels to scales; yet, to effectively binarize the input image, T_1_MS is needed to map scales data to effective thresholds for each pixel. From E_1_IZ and the T_1, T_2, T_3, T_4 images, produced during Step 2, T_1_MS is

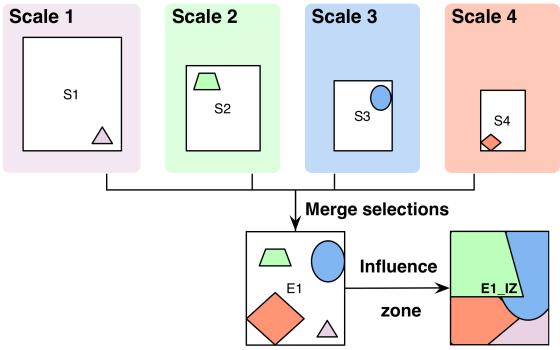


Fig. 5: At the end of Step 2, objects have been assigned to scales. In Step 3 a single image E_1 is built to merge the results. Each object is rescaled to scale 1 if needed and copied to E_1 using their corresponding scale. Some pixels in E_1 are still not mapped to a scale (in white). An influence zone algorithm is used to propagate the scale information to non-mapped pixels and produces E_1_IZ.

deduced: for each pixel p at scale 1, we know its scale $s = E_1_IZ(p)$ and we deduce the corresponding point p' at scale s so that $T_s(p')$ is stored as $T_1_MS(p)$. This image computation is illustrated in Figure 6. At the end of this step, T_1_MS provides a single threshold for each pixel.

Reusing the thresholds in T_s is equivalent to computing the thresholds in I_1 with the window corresponding to scale s. That way, the window is defined pixel-wise which contrasts with the global approach of the original method.

2.6 Step 4: Final binarization

A point-wise binarization is performed with I_1 and $T1_MS$ to get the final result stored in B_1.

3 Optimization

This algorithm is designed to be part of a whole document processing chain. Performance is thus a requirement. The multiscale approach implies extra computation: in addition to a classical implementation of Sauvola's algorithm, we introduce three new steps. Thanks to the multiscale approach, most of the computation is performed on subsampled (smaller) images which limits the impact of the additional steps. Whatever the size of an image, iterations over all its pixels is time consuming, so the main goal of the following optimization is to reduce the number of iterations performed on images. Working at scale 1 is also expensive because of its full resolution. Therefore, step 2 is restricted to scale $s >= 2$ and the original input image is only used to initialize multiscale inputs and to perform the final binarization.

3.1 Step 1: Setup of Input Data

In order to prepare multiscale computations, successive antialiased subsamples are computed. Image at scale s is computed thanks to the image at scale $s-1$ by computing for each pixel at scale s the average value of its neighboring pixels at scale $s-1$. Computing this way reduces the number of operations from $3 \times height_1 \times width_1$ to $height_1 \times width_1 \times (1+\frac{1}{q^2}+\frac{1}{q^4})$, where $height_1$ and $width_1$ are respectively the height and the width of images at scale 1.

Subsampling is performed using integer ratios. Images not having dimensions divisible by these ratios are handled by adding special border data: data located on the inner border of the image are duplicated in an added outer image border. The size of this new border is adjusted to make the dimension of the so extended image a multiple of the subsampling ratio. Having images with an exact proportional size is required to find corresponding pixels between images at different scales.

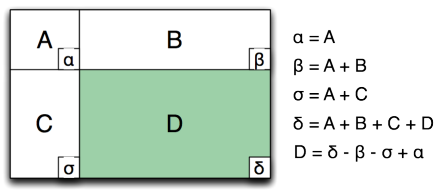


Fig. 8: Reading in an integral image. The integral value of region D is reduced to a simple addition of four values, whatever the size of the window considered.

An integral image is also computed in a single pass. The use of integral images in binarization algorithms was first introduced by Shafait et al. [20] allowing local thresholding methods, such as Sauvola's method, to run in time close to global methods. The idea is to compute an image in which the intensity at a pixel position is equal to the sum of the intensities of all the pixels above and to the left of that position in the original image. Thanks to such an image, as shown in Figure 8,

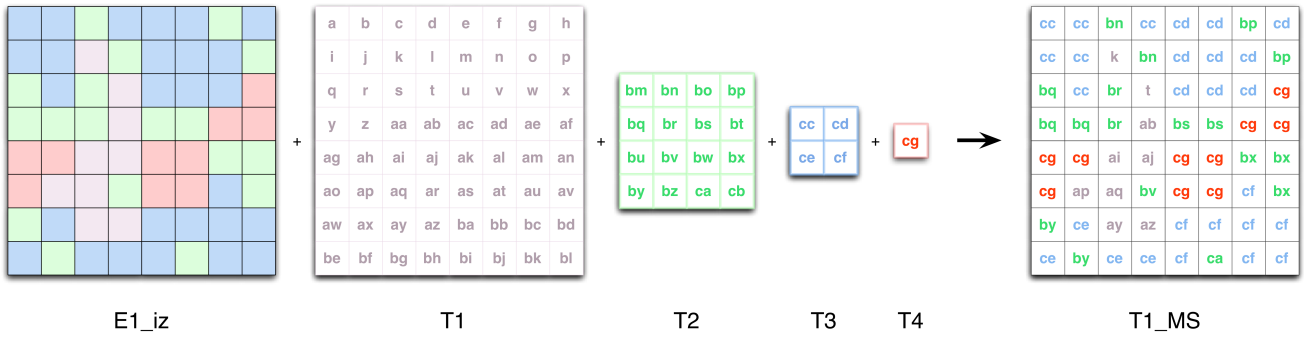


Fig. 6: Selection of the appropriate threshold for final binarization. Here colors corresponds to a scale and letters to pixel values. Each pixel is mapped to a scale in E_1_IZ. Mapping a pixel to its corresponding threshold is performed by reading the threshold image of its associated scale. Each pixel-wise threshold is stored in the resulting image T_1_MS.

computing the mean $m(x, y)$ boils down to:

$$m(x,y) = \frac{1}{w^2} \times (I(x+\frac{w}{2}, y+\frac{w}{2}) + I(x-\frac{w}{2}, y-\frac{w}{2}) - I(x+\frac{w}{2}, y-\frac{w}{2}) - I(x-\frac{w}{2}, y+\frac{w}{2})).$$

Our algorithm uses integral images to compute both the local means and variances which respectively need local sums and squared sums. For performance reasons, these statistics are stored in a single image as a pair: single data block and directional reading enables data locality. A single integral image is computed from I_1 so that statistics are exact. It is stored in an image at scale 2 because it is needed only for scales $s >= 2$ (as explained further) and its reduces the memory footprint. Since there exists a relationship regarding the pixel positions between images at different scales, it is possible to read directly in that image from any scale with the guarantee of computing exact statistics in constant time.

The integral image and the subsampled images are actually computed from the same data and with the same process: iteration over the whole image and computation of sums of values. In our implementation, as shown in Figure 7, the first subsampled image, I_2, and the integral image are computed at the same time. It saves one more iteration on the input image and many floating point operations.

3.2 Step 2: Object Selection at Each Scale

Through this step, each subsampled image is binarized, objects are selected and finally marked in the scale image E_2. In the general approach, each subsampled image needs to be binarized and labeled before selecting objects thanks to an area criterion. These three algorithms follow a common pattern:

Binarization. *For each pixel*, a threshold is computed and defines whether the current pixel is part of an object or of the background.

Labeling. *For each object pixel*, *during a first pass*, performing a backward iteration, a new parent is marked so that, at the end of the pass, pixels of the same object are linked altogether thanks to a parent relationship. *During a second pass*, performing a forward iteration, component labels are propagated within components. *Component area can also be retrieved directly* because of the way relationships were created. This labeling is based on Tarjan's Union-Find algorithm [25].

Filtering. *For each object pixel*, if the object area is too low or too high, the pixel is considered as belonging to background.

Marking. *For each selected object pixel*, mark the pixel in E_2 with the current scale as value.

Thanks to this pattern, these steps can be combined into a single algorithm, described in 1, decreasing the number of iterations on the whole image from five down to two.

In 1, seven images are used: I_s, the subsampled input image, T_s, the image storing thresholds used for binarization, $Parent_s$, storing the parent relationship for the Union-Find algorithm, $Card_s$, storing the components' area, B_s, the filtered and binarized image, E_2, the image at scale 2 where retrieved objects are marked with their corresponding scale, and Int_2, the integral image at scale 2. At any time, for a specific pixel in I_s,

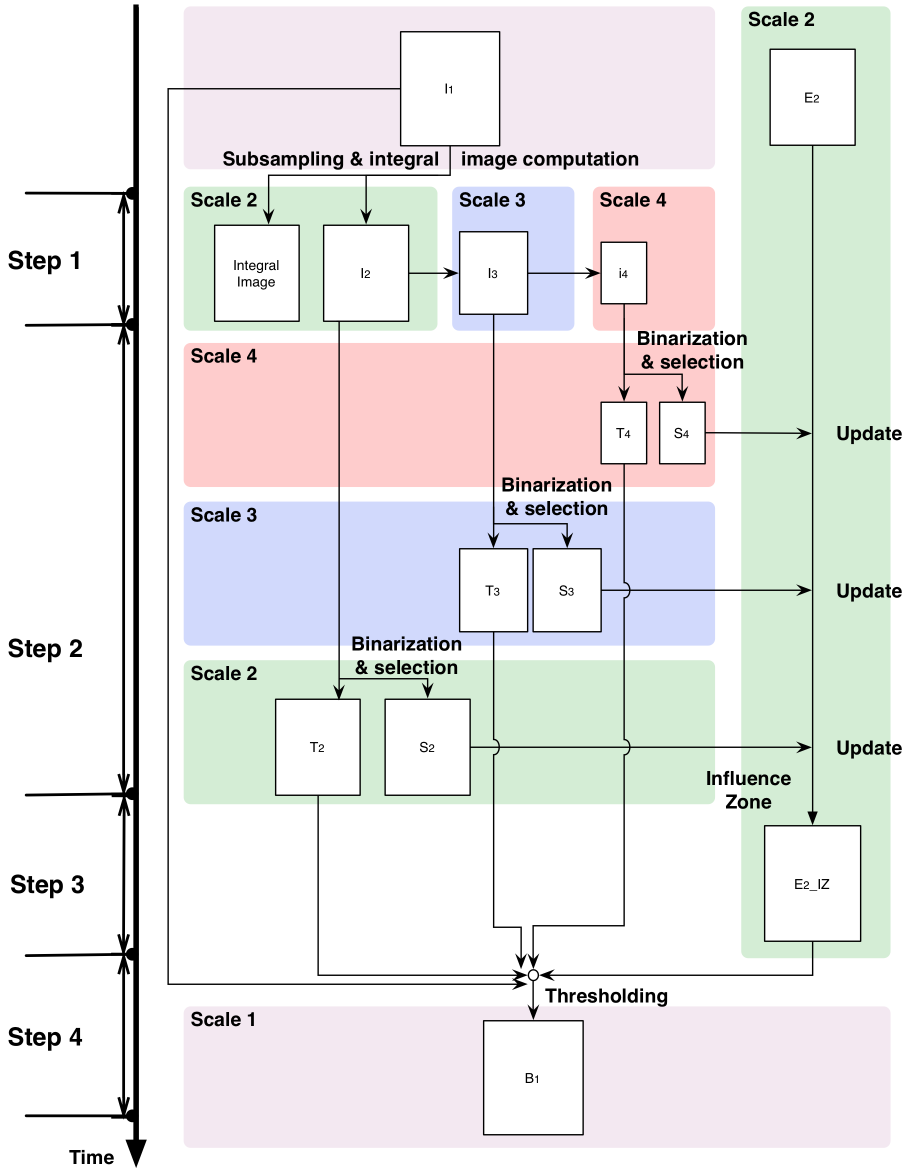


Fig. 7: Full optimal data workflow of the proposed method. It describes how the method is effectively implemented and computes the result.

the corresponding data must be readable in all these images at different scales. This algorithm is composed of two passes. In the first pass, the input image at scale s is binarized and the component area is computed. In the second pass, components are filtered with the area criterion and marked in the scale image E_2.

In the first pass, for each pixel, statistics are computed from the integral image Int_2 (line 14). Note that in compute_stats(), the window size effectively used to read Int_2 is actually of size $w_{2,s}$ (see subsection 2.4). Then Sauvola threshold is computed (line 15) and the pixel value is thresholded (line 16). If the thresholded value returns True, either a new component is created or the pixel is attached to an existing component in its neighborhood. In those both cases, component area is updated (lines 19 and 24).

In the second pass, all pixels belonging to a component with an area within the valid range are marked in E_2 (line 32). Note that those pixels are marked in E_2 with the current scale as value using mark_pixel_in_scale_image() but only if they have not been marked before (lines 34

and 39). Processing these selected pixels is straightforward and does not require any labeling.

Algorithm 1 Step 2 algorithm - Union-Find based algorithm combining Sauvola's binarization and object selection.

```
// Input
integer s                // Current scale
image2d I_s              // Grayscale Input image at scale s
image2d Int_2            // Integral image at scale 2
image2d E_2              // Scale image at scale 2
integer w                // Window width

// Local variables
image2d T_s              // Image of thresholds
image2d B_s              // Binary image of I_s
image2d Card_s           // Image of connected component
                         // cardinality
image2d Parent_s         // Image of pixel's parent relation-
                         // ship
// First pass
for all pixel p in I_s do
    (mean,stddev) ← compute_stats(Int_2, w)
    T_s(p) ← sauvola_threshold(I_s(p), mean, stddev)
    B_s(p) ← I_s(p) < T_s(p)
    if B_s(p) = true then // p Is part of an object
        do_union(Parent, p, p)
        update_component_area(p)
    end if
    for all neighbor n of p do
        if B_s(n) = true then // n is part of an object
            do_union(Parent, p,n)
            update_component_area(Card_s, n)
        end if
    end for
end for

// Second pass
for all pixel p in backward order in I_s do
    if B_s(p) = true then // p is part of an object
        if is_root(Parent_s, p) then
            B_s(p) ← is_component_area_valid(Card_s,
I_s(p))
            if B_s(p) = True and is_not_marked_in-
_scale_image(E_2, p) then
                mark_pixel_in_scale_image(E_2, p, s)
            end if
        else// Propagation
            B_s(p) ← B_s(Parent_s(p))
            if B_s(p) = True and is_not_marked_in-
_scale_image(E_2, p) then
                mark_pixel_in_scale_image(E_2, p, s)
            end if
        end if
    end if
end for
```

At the end of this step, four images are available for the next steps: E_2 and T_2, T_3, and T_4. Note that since this step is performed on scales $s >= 2$, the scale image is only known at scale 2 (not at full scale) and there are only three T_s threshold images instead of 4. Therefore, avoiding computation at scale 1 reduces memory usage and saves some execution time.

3.3 Step 3: Result Merging

Since E_2 is built iteratively during step 2, no merging is needed anymore here. Only the influence zone is performed on E_2 to produce E_2_IZ.

3.4 Step 4: Final Binarization

During this step, the aim is to compute a binarization of the input image I_1. Images T_2, T_3, T_4, E_2_IZ, I_1 and the output image are browsed simultaneously. The process remains identical to the one depicted in Figure 6 except that the threshold image T_1_MS is never created: the binarization is performed directly once the threshold is found.

To prevent many memory accesses because of the numerous images to read, we rely on the scale relationship and iterate over the pixels of all the images simultaneously. We rely on the Morton order [26] to iterate over pixels in square subparts of images. In Figure 9 reading a square of four pixels in the left image is equivalent to read a single pixel in the two other images. A new value is read in subsampled images only if all the pixels corresponding to the current one have been processed. Such an iteration in this final binarization reduces the total number of memory accesses in the images from $6 \times height_1 \times width_1$ to less than $3 \times height_1 \times width_1$ where $height_1$ and $width_1$ are respectively the height and the width of images at scale 1.

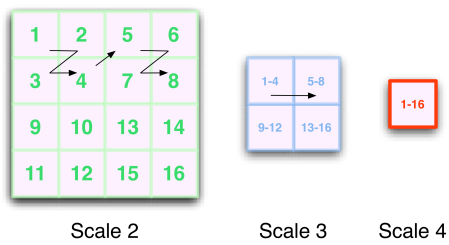


Fig. 9: Reduction of memory accesses thanks to the Morton order while reading threshold images. At scale 2 image values are accessed 15 times while they are accessed 3 times at scale 3 and only once at scale 4.

4 Experimental Results

4.1 Analysis of the method

During the development of this method, we have used the document dataset of the page segmentation competition from ICDAR 2009[27]. It is composed of 63 A4 full page documents at 300dpi. It contains a mix of magazines, scientific publications and technical articles. We run our multiscale algorithm with several window values on that dataset and found that, $w = 51$ gives good results in all cases. This is the reason why we use that value in the following tests and evaluation.

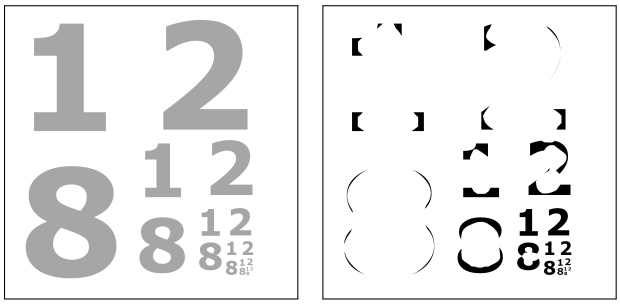
10a is a good candidate to illustrate the drawbacks of the original strategy of Sauvola's algorithm. The original size is 1500×1500 pixels and different sizes of objects are represented: the largest is more than 370 pixels and the smallest around 10 pixels tall. Object thickness varies from 40 to 1 pixels. Running Sauvola's algorithm leads to almost empty incomplete binarized objects (10b). 10c shows that the multiscale implementation takes object sizes into account. Here objects are colored according to the scale they have been retrieved from: green for scale 2, blue for scale 3 and red for scale 4. It clearly shows the dispatch of the different object sizes into the different scales. As a consequence Large objects are clearly well retrieved (10d).

Figure 11 also shows some examples of the binarization results performed with this method on real documents. One can also notice the limits of using the object area as criterion: in 11a thick line separators are retrieved at scale 3 but they should be retrieved at scale 2 for best results.

Table 1 shows computation times results obtained on an Intel Xeon W3520@2,67Ghz with 6 GB of RAM and programs compiled with GCC 4.4 with O3 optimization flag. While the classical Sauvola's algorithm runs in 0.05s on A4 document images scanned at 300dpi, the multiscale implementation runs in 0.15s. Table 2 illustrates in details this difference on a larger image. As expected, multiscale features implies a cost on computation time: it is about 2.45 times slower than the classical implementation on large images mainly due to the multiscale processing. The computation is constant with respect to the input image size.

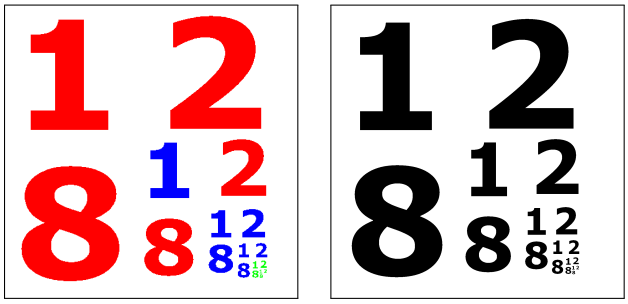
4.2 Adjustment of Parameter k

In the original binarization method proposed by Sauvola, the parameter k is set globally for the whole document. Adjusting this parameter can lead to better results on documents with low contrast or thin characters. In the multiscale approach, it is possible to set a different k value for each scale, e.g., for ranges of object sizes. We have compared such a variant with the classical (monoscale) approach where k is set to 0.34 globally. We will notate k_s the value of parameter k at scale s. According to our experiment using $k_2 = 0.2$, $k_3 = 0.3$ and $k_4 = 0.5$ gives good results. At scale 2 and 3, only small and medium objects are retrieved. They can be thin or not contrasted enough, so setting a low value of k_2 and k_3 allows us to be less strict in Sauvola's formula, i.e retrieving pixels with lower con-



(a) Original image. (1500 × 1500 pixels)

(b) Result with Sauvola's original algorithm.



(c) Scale image used in Sauvola multiscale. Each color corresponds to a scale.

(d) Result with Sauvola multiscale.

Fig. 10: Result improvements. The input image contains several objects of different sizes. Therefore a single window size successfully retrieves every objects at the same time. The multiscale version acts just like it adapts the window size to the image local contents so that thresholds are relevant and objects are correctly identified.

Image size	1500 × 1500	2300 × 3150	7780 × 11600
Sauvola	0.01s	0.05s	0.85s
Multiscale Sauvola	0.05s	0.15s	2.08s

Table 1: Comparison of Sauvola's implementation computing times.

(a) (b)

Fig. 11: Examples of scale maps produced at the end of Step 2. The multiscale algorithm detects the different sizes of objects: core text/small objects in green, subtitles/medium size objects in blue and titles/large objects in red.

Step	Time
Integral image computation	0.26s
Binarization	0.59s
Total	0.85s

(a) Classical Sauvola's algorithm.

Step	Time
1 - Integral image computation & subsamplings	0.45s
2 - Multiscale processing	0.87s
3 - Influence zone on scale image	0.28s
4 - Final binarization	0.48s
Total	2.08s

(b) Multiscale Sauvola's algorithm.

Table 2: Computation time details between two implementations with an image of 7780×11600 pixels as input. The multiscale implementation introduces several extra steps inducing some performance costs.

trasts. At scale 4, large objects are retrieved and they are large enough not to need some additional precision. This new version, with one value of k per scale, namely Sauvola MS_{kx}, is evaluated in the next section, along with other methods.

5 Evaluation

All the material used in this section (datasets, ground truths, implementations and benchmark tools) are freely available online from the resources page related to this article[1].

The required quality of the binarization highly depends on use cases. We have chosen to evaluate two aspects which are important in a whole document analysis toolchain: pixel-based accuracy and *a posteriori* OCR-based results.

The evaluation is performed with eight binarization algorithms. We propose two implementations of Sauvola Multiscale: one with a fixed k value, Sauvola MS_k, and another one where k is adjusted according to the scale, Sauvola MS_{kx}. We compare those two implementations to the classical (monoscale) algorithm from Sauvola and height other state of the art methods: Wolf's [21], Otsu's [2], Niblack's [3], Kim's [22], TMMS [28], Sauvola MsGb [10], Su 2001 [29], and Lelore 2011 [14].

TMMS [28] is a morphological algorithm based on the morphological toggle mapping operator [30]. The morphological erosion and dilation are computed. Then, if

[1] `http://publis.lrde.epita.fr/201302-IJDAR`

Method	Parameter values
Sauvola	$w = 51$ and $k = 0.34$
Sauvola MS_{kx}	$w = 51$ $k_2 = 0.2, k_3 = 0.3, k_4 = 0.5$
Sauvola MS_k	$w = 51$ and $k = 0.34$
Wolf	$w = 51$ and $k = 0.34$
Otsu	N.A.
Niblack	$w = 51$ and $k = -0.2$
Kim	$w = 101$ and $k = 0.34$
TMMS	$cminL = 20$, $cminH = 45$ and $p = 30$
Sauvola MsGb	N.A.
Su 2011	N.A.
Lelore 2011	N.A.

Table 3: Parameters used to evaluate each method.

the pixel value is closer to the erosion value than to the dilation value, it is marked as background, otherwise it is marked as foreground. If this method was initially dedicated to natural images, HDIBCO 2009 challenge [16] shows that this algorithm gives also good results on document images. It was ranked $2nd$ out of 43.

Multiscale Grid-based Sauvola (Sauvola MsGb), introduced by Farrahi Moghaddam and Cheriet [10], is a multiscale adaptive binarization method based on Sauvola formula and a grid-based approach. This method was initially dedicated to binarize degraded historical documents while preserving weak connections and stroke. It was ranked $9th$ out of 15 at HDIBCO 2010 and $14th$ out of 24 at HDIBCO 2012.

Su 2011 [29], relies on image contrast defined by the local image maximum and minimum. Text is then segmented using local thresholds that are estimated from the detected high contrast pixels within a local neighborhood window. This method was initially dedicated to processing historical documents. It was ranked $2nd$ out of 18 at DIBCO 2011.

Lelore and Bouchara's method [14, 15] is based on the one which won DIBCO 2011 but without upscaling the input images. In a first step, the method achieved a rough localization of the text using an edge-detection algorithm based on a modified version of the well-known Canny method. From the previous result, pixels in the immediate vicinity of edges are labeled as text or background thanks to a clustering algorithm while the remaining pixels are temporarily labeled as unknown. Finally, a post-processing step assigns a class to these 'unknown' pixels.

For all methods, we have chosen the best parameter values. The parameter k, has been set according to the recommended values in the literature. Concerning the window size w, we have run the algorithms on the document dataset of the page segmentation competition (PSCOMP) from ICDAR 2009 [27] and we have tuned the value to obtain results with good visual quality.

TMMS parameters have been set by its author based on the results on the PSCOMP dataset. Su 2011 is self-parameterized and so is Sauvola MsGb since we relied on its *smart mode*. All parameter values are summarized in Table 3. For each method, the meaning of parameters is detailed in their respective reference article. Note that for TMMS, as the author did for DIBCO challenge, an hysteresis is used to set up the cmin parameter; that is the reason why there are two values for it. It is important to note that parameters are fixed for the whole evaluation in order to highlight the robustness of the methods.

For most methods, we used their freely accessible implementations. We implemented Kim's technique. TMMS, Sauvola MsGb and Su 2011 were provided as binaries by their respective authors. Binarization results for Lelore's method have been directly computed by its author.

The evaluation is performed on two document types: historical documents and magazines.

5.1 Datasets

As shown in Table 4, the most common reference datasets for document image analysis provide pieces of documents without the whole context data. A majority of them, DIBCO's and HDIBCO's documents, are dedicated to historical documents, which include also handwritten documents. It implies specific cares for reconstructing or preserving thin characters, and for separating background and foreground data which may require special processing before and/or after binarization to obtain best results. Prima Layout Analysis Dataset (LAD) is composed of more than 300 full page documents from magazines and articles with a good quality but it only provides the layout ground truth.

Our method remains a raw binarization method, without any pre- or post-processing, and is designed to work best on magazines with less severe degradations than in DIBCO datasets.

For the next evaluation, we use DIBCO 2009 and 2011, and HDIBCO 2010 and 2012 datasets for historical documents, and our own datasets, LRDE Document Binarization Dataset (DBD) for magazines.

LRDE DBD is composed of two subsets. Both are based on an original vector-based version of a French magazine. From this magazine, we have selected 125 A4 300-dpi pages with different text sizes.

One subset of images has been directly extracted from the digital magazine and rasterized. Those images are clean with no illumination nor noise issues.

	PRImA LAD[31]	HDIBCO 2012[32]	DIBCO 2011[18]	HDIBCO 2010[17]	DIBCO 2009[16]	CMATER DB6[33]	**LRDE DBD**
Number of documents							
Full Pages (A4)	305	-	-	-	-	-	**375**
Piece of pages	-	14	16	10	10	5	**-**
Type of documents							
Magazines	X	-	-	-	-	-	**X**
Technical articles	X	-	-	-	-	X	**-**
Business publications	X	-	-	-	-	X	**-**
Technical publications	X	-	-	-	-	-	**-**
Historical	-	X	X	X	X	-	**-**
Historical handwritten	-	X	-	X	-	-	**-**
Quality							
Degraded	-	14	16	10	10	5	**-**
Scanned	305	-	-	-	-	-	**125**
Digital	-	-	-	-	-	-	**250**
Ground truth							
Binarization	-	14	16	10	10	5	**125**
Layout	305	-	-	-	-	-	**-**
OCR output	-	-	-	-	-	-	**375**

Table 4: Overview of the reference datasets in document image analysis. Despite a wide variety of datasets, none of them provide binarization ground truth for full documents or OCR output. Our dataset LRDE *Document Binarization Dataset (*DBD*)* is meant to fulfill this need.

For every pages, we have removed the pictures in order to make the groundtruthing and the evaluation easier. Our selection includes pages with background colored boxes and low contrasts. This dataset is used both for the pixel-based accuracy and the OCR-based evaluation. We will refer to it as the *clean documents* dataset.

The other subset is based on the same documents that have been first printed as two-sided pages then scanned at 300-dpi resolution. A rigid transform has been applied to each document so that text lines match the ones of the corresponding *clean documents*. Therefore this process has introduced some noise, show-through, and illumination variations. This subset is used for OCR-based evaluation only. We will refer to it as the *scanned documents* dataset.

For each page, text lines have been grouped into three categories w.r.t. their font size. Lines with a x-height less than 30 pixels are categorized as *Small (S)* and correspond to core paragraph text; lines with a x-height between 30 and 55 pixels are considered as *Medium (M)* and correspond to subtitles or small titles; and for higher x-height, lines are considered as *Large (L)*, e.g., titles. A set of lines composed of 123 large lines, 320 medium lines and 9551 small lines is available for both *clean documents* and *scanned documents*. For the OCR-based evaluation, we are thus able to measure the binarization quality for each text category independently.

The ground truth images have been obtained using a semi-automatic process. To that aim we rely on a binarization using a global threshold. Sometimes, due to contrast or color issues, some objects were not correctly binarized or were missing in the output, therefore we made some adjustments in the input image to preserve every objects.

In order to produce the OCR ground truth, we used Tesseract 3.02 [34] on the *clean documents* dataset. Errors and missed text were fixed and text was grouped by lines to produce a plain text OCR output reference.

Datasets, associated ground truths, implementations, and the whole set of tools used for this evaluation are freely available online from the resources page related to this article[2].

5.2 Evaluation with Historical Documents

We have tested our method on DIBCO/HDIBCO datasets from 2009 to 2012 [18, 32, 18, 17] with the parameters given in Table 3. The results are detailed in Table 5. On historical documents the multiscale versions of Sauvola are roughly comparable to the original Sauvola's method. That was expected since the historical documents of the contest databases do *not* contain "multiscale text"; text is only either of small or of medium size. Note that the original Sauvola method (or any of its variations) is a monolithic general-purpose binarization algorithm. It cannot compete with some elaborate binarization chains, including pre- and post-processings, and *a fortiori* dedicated for historical documents.

All output images are available on the web page related to this article for extra analysis.

5.3 Evaluation with Magazines

5.3.1 Pixel-based Accuracy Evaluation

The *clean documents* dataset is used for pixel-based accuracy evaluation since the ground truth binarized documents are perfectly known.

Evaluation Measure. According to common evaluation protocols [18] we used the F-measure (FM) in order to compare our method with other approaches:

$$FM = \frac{2 \times Recall \times Precision}{Recall + Precision},$$

[2] `http://publis.lrde.epita.fr/201302-IJDAR`

	Method	FM	p-FM	PSNR	DRD	MPM
2009	Sauvola	85.04	-	16.83	5.22	1.02
	Sauvola MS_k	78.08	-	15.24	14.08	4.51
	Sauvola MS_{kx}	76.85	-	14.52	18.52	8.97
2010	Sauvola	59.86	68.29	14.73	9.35	1.32
	Sauvola MS_k	61.17	69.45	14.72	9.51	1.82
	Sauvola MS_{kx}	80.03	87.07	16.36	6.90	3.42
2011	Sauvola	81.06	-	15.90	5.98	4.84
	Sauvola MS_k	79.31	-	15.33	8.16	11.54
	Sauvola MS_{kx}	79.70	-	14.91	11.67	20.44
2012	Sauvola	69.76	75.08	15.91	8.65	1.36
	Sauvola MS_k	69.56	74.78	15.12	10.37	2.79
	Sauvola MS_{kx}	81.77	86.41	16.51	8.37	5.34

Table 5: Pixel-based accuracy evaluation results performed on the DIBCO/HDIBCO subsets from 2009 to 2012.

Method	Precision	Recall	FM	Time (s)
Sauvola MS_{kx}	0.97	**0.94**	**95.0**	170
Lelore	0.99	0.88	92.9	1625
Sauvola MS_k	0.97	0.89	92.1	170
TMMS	0.90	**0.95**	92.0	250
Wolf	0.99	0.85	91.4	125
Otsu	0.98	0.84	90.3	67
Sauvola	0.99	0.82	89.7	155
Kim	0.99	0.82	89.3	260
Sauvola MsGb	0.99	0.82	89.3	111600
Niblack	0.89	0.91	88.8	95
Su 2011	0.98	0.80	87.3	8800

Table 6: Evaluation of the binarization results of different methods over 125 documents extracted from a digital issue of a French magazine. Binarization results are compared to the ground truth. The time needed to run each algorithm over the 125 images is given with input/output overhead included.

where $Recall = \frac{TP}{TP+FN}$ and $Precision = \frac{TP}{TP+FP}$, with TP, FP, and FN respectively standing for true-positive (total number of well-classified foreground pixels), false-positive (total number of misclassified foreground pixels in binarization results compared to ground truth), and false-negative (total number of misclassified background pixels). In tables, the F-measure is expressed in percentage.

Results and Analysis. Table 6 gives the evaluation results.

A selection of three regions of document images are depicted in Figure 12, 13 and 14 to compare the results of the different methods. We can see that our approach increases the result quality of the classical binarization method proposed by Sauvola by five percentage points. This difference is mainly due to a now-adapted window size that can adequately process large objects, as we can see on Figure 12 and Figure 14. Thanks to the multiscale approach, locally low-contrasted objects may be retrieved because they are considered on a larger area. This is the case in Figure 13 where there is a large and low-contrasted object.

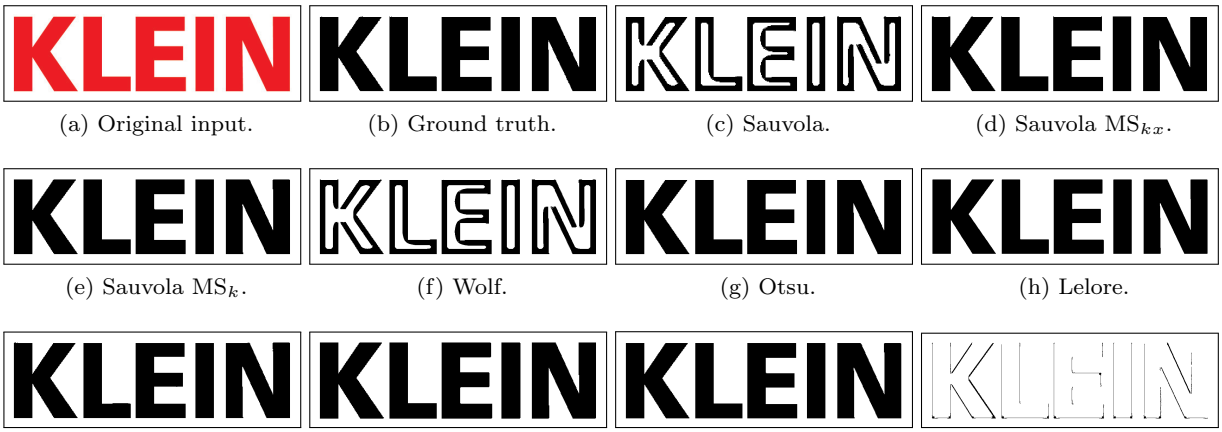
Compared to Sauvola's approach, Sauvola MS_k and Sauvola MS_{kx} are able to retrieve the right part of the object. Niblack's method is able to find it but the too small window prevents it from retrieving it completely. Wolf's method performs relatively well but some objects are missing in the output. Otsu's method performs better than any Sauvola's approach. This is understandable because it is known to give good results on clean documents, which is the case here, and can retrieve large objects correctly. Its corresponding score results mainly from missing objects because of low contrasts (see 13g). Niblack performs well in the text but does not handle color text boxes edges correctly. Transitions between color boxes and the background lead to some artifacts. Same issues arise with textured background. Kim encounters some trouble with text in colored text box which leads to large artifacts, the box being considered as object instead of as background. Sauvola MsGb performs as well as Sauvola and surprisingly has some difficulties to extract large text like titles and drop capitals. This is also the case for Su's method: large text and large reverse video areas are missing or degraded (12l and 13l). In addition, small text edges are not as smooth as they should be. Lelore's method gives good overall results, although it has some difficulties some times on large titles and small text. Its performances are really close to ours.

In Table 6 the time needed to compute the final binarization results on 125 documents confirms the multiscale overheads as compared to the classical Sauvola's implementation. It shows also the large range of computation times, which is a crucial information to take into account while choosing an algorithm.

This evaluation also shows that adjusting the parameter k w.r.t. the scale, in our approach, may improve the quality of the results. Sauvola MS_{kx} gets a three points higher F-measure than Sauvola MS_k. Those results highlight that, despite the most recent binarization methods perform very well on historical documents, they may not be able to properly binarize simple and clean magazine document images.

5.3.2 OCR-*based Evaluation*

Evaluation Method. Once a document has been binarized, and character recognition is performed, line by line, by Tesseract (the options: `-l fra -psm 7` specify the recognized language, French, and that the given

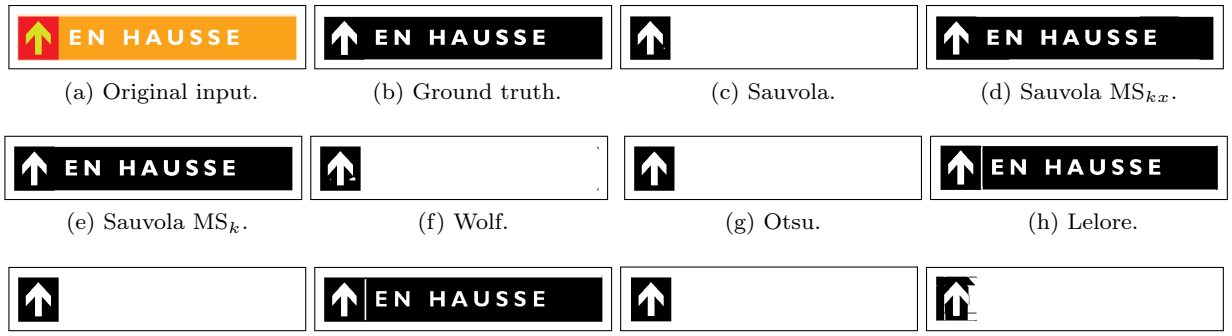


(a) Original input. (b) Ground truth. (c) Sauvola. (d) Sauvola MS_{kx}.

(e) Sauvola MS_k. (f) Wolf. (g) Otsu. (h) Lelore.

(i) Kim. (j) TMMS. (k) Sauvola MsGb. (l) Su 2011.

Fig. 12: Comparison of binarization on large text (images of 1034×290 pixels extracted from *clean documents*, page 114). As expected, statistics-based methods are not robust enough to handle large objects with a fixed window size. Results have been computed on the full document page.

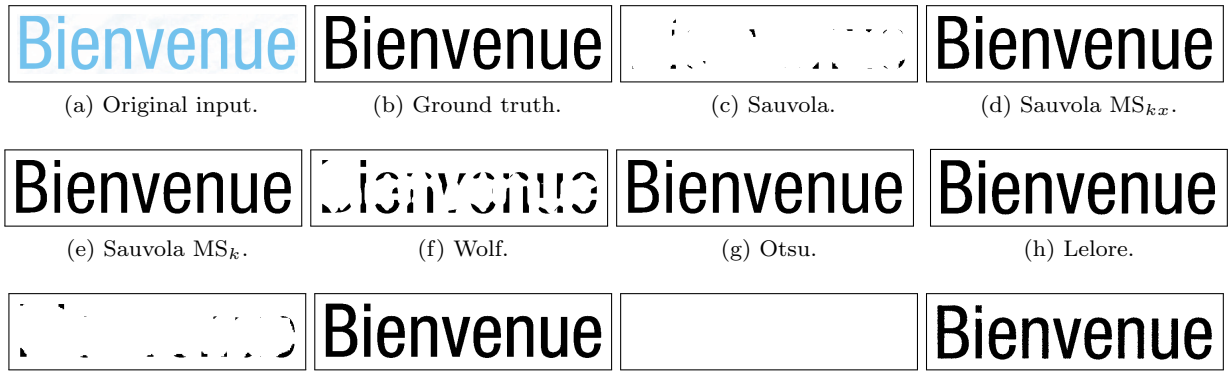


(a) Original input. (b) Ground truth. (c) Sauvola. (d) Sauvola MS_{kx}.

(e) Sauvola MS_k. (f) Wolf. (g) Otsu. (h) Lelore.

(i) Kim. (j) TMMS. (k) Sauvola MsGb. (l) Su 2011.

Fig. 13: Comparison of binarization on large cartouche with reverse video and low contrast between foreground and background (images of 702×116 pixels extracted from *clean documents*, page 123). Results have been computed on the full document page.



(a) Original input. (b) Ground truth. (c) Sauvola. (d) Sauvola MS_{kx}.

(e) Sauvola MS_k. (f) Wolf. (g) Otsu. (h) Lelore.

(i) Kim. (j) TMMS. (k) Sauvola MsGb. (l) Su 2011.

Fig. 14: Comparison of binarization of large text with a textured background (images of 704×149 pixels extracted from *clean documents*, page 187). Results have been computed on the full document page.

image contains only one text line). The OCR has been run on the two datasets: *clean* and *scanned documents*, and the error is computed thanks to the Levenshtein distance.

Results and Analysis. Table 7 shows the recognition rate for seven binarization algorithms w.r.t. text line quality and x-height.

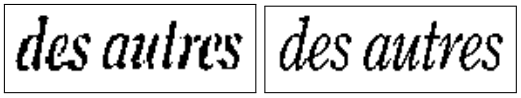
Our Sauvola MS_{kx} and Sauvola MS_k implementations almost always outperformed the original Sauvola's algorithm and some state-of-the-art algorithms. On clean documents results are very close thanks to clean backgrounds, which dramatically reduce the risk of erroneous retrievals. The small differences that are encountered here are due to the contrast between the text and the background colors: text is usually well retrieved but edges are not always as clean as those obtained with a multiscale Sauvola's implementation (see Figure 16). This fact is even more true for scanned documents where illumination variations and show-through artifacts are introduced. Sauvola MS_k and Sauvola MS_{kx} performs better than the classical Sauvola's algorithm on large text, e.g., text with a high x-height. It is globally more robust and the results are more stable on a wider set of objects. Moreover they do not need fine parameter adjustment to deliver acceptable results with any object sizes.

Regarding TMMS method, the results are globally equivalent to Sauvolas algorithm for clean documents. On scanned documents, the text is correctly binarized but cartouches and colored text boxes are considered as foreground (see Figure 15). They surround text components, preventing the OCR from recognizing the text correctly. Since this is a common situation in this dataset, the OCR error is thus extremely high compared to the other methods. To be usable in our context, this method would require post-processing. Su's method does not scale to large objects thus giving really bad results for that category. Results quality for small and medium text are below average due to many broken and missing characters. This method seems to be really specific to the kind of noise it was designed for. Surprisingly, Otsu's method performs relatively well on both clean and scanned documents despite a non-uniform illumination. Lelore's method performs very well on clean documents, but, on scanned documents. Small characters are usually broken and large ones have holes.

Except for Otsu's and Lelore's methods, the main drawback of state-of-the-art methods is their difficulty to correctly binarize large objects.

(a) Input image. (b) TMMS. (c) Sauvola MS_k.

Fig. 15: Comparison of binarizations on cartouches. TMMS considers both text and cartouches as foreground on scanned documents which causes high OCR error.



(a) Wolf. (b) Sauvola MS_k.

Fig. 16: Comparison of binarization methods on small text (x-height of 20 pixels). Edges are not always as well retrieved as with Sauvola MS_k because of low contrast between text and background.

Method	Raw OCR error (%)					
Set →	Clean documents			Scanned documents		
Subset →	S	M	L	S	M	L
Sauvola	2.62	2.61	6.00	5.49	3.87	7.75
Sauvola MS_{kx}	2.59	2.21	4.83	5.14	2.74	5.68
Sauvola MS_k	2.64	2.60	4.78	5.44	3.20	5.15
wolf	2.60	2.42	5.04	5.14	3.43	6.53
Otsu	3.09	2.55	4.56	6.23	3.58	5.73
Niblack	2.68	2.28	6.79	4.96	5.15	12.79
Kim	2.79	3.01	5.47	7.03	5.08	7.80
TMMS	2.61	2.43	5.25	18.17	11.44	54.83
Sauvola MsGb	5.45	5.14	9.29	9.49	8.40	10.35
Su 2011	2.95	5.01	15.39	7.42*	8.54*	31.58
Lelore	2.46	2.21	4.88	8.01	3.44	8.65

*Due to program crashes, those scores do not include the results of 2 lines out of 320, for scanned medium text, and of 57 lines out of 9551, for scanned small text.

Table 7: Raw OCR error rates of different binarization algorithms on the provided dataset according to the text size (S/M/L stand respectively for small/medium/large) . Best scores (e.g., lowest values), are in a bold typeface.

6 Reproducible Research

We advocate *reproducible research* [35], which means that research product shall also include all the materials needed to reproduce research results. To that aim we provide the community with many resources related to this present article, available from `http://publis.lrde.epita.fr/201302-IJDAR`.

First a demo is online, so that the user can upload an image and get the binarization result of our method; therefore third-party people can quickly reproduce our results or run some extra experiments on their own data. Second we also provide an implementation of our method. It is part of the SCRIBO module [36] of the Olena project [37], along with most of the algorithms discussed in this paper. The Olena project is an open source platform dedicated to image processing, freely available from our website[3]. It contains a general-purpose and generic C++ image processing library, described in [38]. Last the magazine document image database (including ground truths and the results of 10 binarization methods, see Table 4) that we have set up is now usable by the community.

As a matter of comparison, among the 36 methods that entered the DIBCO 2011 and HDIBCO 2012 competitions, an implementation is available for only two of them and 31 of them cannot be reproduced due to more or less partial descriptions or missing parameter settings.

7 Conclusion and Future Work

In this paper we propose an approach that significantly improves the results of Sauvola's binarization on documents with objects of various sizes like in magazines. Sauvola's binarization is made almost insensitive to the window parameter thanks to this implementation.

Its accuracy is tested on 125 300-dpi A4 documents. Where on small and medium text sizes this implementation gets better or similar results than the classical implementation, it dramatically improves the results for large text in magazines. This property is very important for document analysis because text using large font sizes usually correspond to titles and may be particularly relevant for indexing purpose. Furthermore, pixel-based accuracy and character recognition rates are also improved by our proposal. That is crucial for a whole document analysis, from the layout to the contents. Sauvola's formula is probably not the best one to use for historical documents but at least our evaluation showed that it still competes with the latest awarded methods regarding magazines and classical documents. We also proposed a fast implementation of our method, limiting the impact of the additional steps to a 3 times slower method instead of a 7-times slowdown in a naive version.

The proposed implementation is part of the SCRIBO [36] module from the Olena platform [38], an open-source platform for image processing written in C++, freely available on our website. The SCRIBO module also contains the implementation of some algorithms presented in this paper. An online demo of our method is available[4] where documents can be uploaded for testing purpose.

An issue remains though and may be considered for further investigations. The area criterion used to select at which scale an object should be retrieved is probably not precise enough to make a distinction between large thin objects and large thick objects.

Acknowledgements The authors would like to thank Yongchao Xu, Jonathan Fabrizio and Roland Levillain for proofreading and commenting on the paper, and Reza Farrahi Moghaddam, Thibault Lelore and Frédéric Bouchara for their active collaboration. The authors are very grateful to Yan Gilbert who has accepted that we use and publish as data some pages from the French magazine "Le Nouvel Obs" (issue 2402, November 18th-24th, 2010) for our experiments.

References

1. M. Sezgin and B. Sankur. Survey over image thresholding techniques and quantitative performance evaluation. *Journal of Electronic Imaging*, 13:146–165, 2004.
2. N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, January 1979.
3. W. Niblack. *An Introduction to Digital Image Processing.* Strandberg Publishing Company, 1985.
4. J. Sauvola and M. Pietikainen. Adaptive document image binarization. *Pattern Recognition*, 33:225–236, 2000.
5. E. Badekas and N. Papamarkos. Automatic evaluation of document binarization results. *Progress in Pattern Recognition, Image Analysis and Applications*, pages 1005–1014, 2005.
6. J. Bernsen. Dynamic thresholding of grey-level images. In *Proceedings of the International Conference on Pattern Recognition*, pages 1251–1255, 1986.
7. E. Gabarra and A. Tabbone. Combining global and local threshold to binarize document of images. In *Pattern Recognition and Image Analysis*, volume 3523 of *LNCS*, pages 173–186. Springer, 2005.
8. Y. Rangoni, F. Shafait, and T. M. Breuel. OCR based thresholding. In *Proceedings of IAPR Conference on Machine Vision Applications*, pages 98–101, 2009.
9. S. Tabbone and L. Wendling. Multi-scale binarization of images. *Pattern Recognition Letters*, 24(1–3):403–411, 2003.
10. R. Farrahi Moghaddam and M. Cheriet. A multi-scale framework for adaptive binarization of degraded document images. *Pattern Recognition*, 43(6):2186–2198, 2010.
11. F. Chang, K.-H. Liang, T.-M. Tan, and W.-L. Hwang. Binarization of document images using hadamard multiresolution analysis. In *Proc. of Intl. Conference on Document Analysis and Recognition*, pages 157–160, 1999.
12. S.S. Bukhari, F. Shafait, and T. Breuel. Foreground-background regions guided binarization of camera-captured document images. In *Proceedings of the Inter-*

[3] `http://olena.lrde.epita.fr`

[4] `http://publis.lrde.epita.fr/201302-IJDAR`

national Workshop on Camera Based Document Analysis and Recognition, 7 2009.
13. S. Lu, B. Su, and C. Tan. Document image binarization using background estimation and stroke edges. *International Journal on Document Analysis and Recognition*, 13:303–314, 2010.
14. T. Lelore and F. Bouchara. Super-resolved binarization of text based on the FAIR algorithm. In *Proceedings of International Conference on Document Analysis and Recognition*, pages 839–843, sep 2011.
15. T. Lelore and F. Bouchara. FAIR: a fast algorithm for document image restoration, 2013. to be published.
16. B. Gatos, K. Ntirogiannis, and I. Pratikakis. ICDAR 2009 document image binarization contest (DIBCO). In *Proceedings of ICDAR*, pages 1375–1382, 2009.
17. I. Pratikakis, B. Gatos, and K. Ntirogiannis. H-DIBCO 2010 - handwritten document image binarization competition. In *Proc. of International Conference on Frontiers in Handwriting Recognition*, pages 727–732, 2010.
18. I. Pratikakis, B. Gatos, and K. Ntirogiannis. ICDAR 2011 document image binarization contest (DIBCO). In *Proceedings of International Conference on Document Analysis and Recognition*, pages 1506–1510, 2011.
19. N. Howe. A laplacian energy for document binarization. In *Proceedings of International Conference on Document Analysis and Recognition*, pages 6–10. IEEE, 2011.
20. F. Shafait, D. Keysers, and T. M. Breuel. Efficient implementation of local adaptive thresholding techniques using integral images. *Document Recognition and Retrieval XV*, Jan 2008.
21. C. Wolf and J.-M. Jolion. Extraction and recognition of artificial text in multimedia documents. *Pattern Analysis & Applications*, 6:309–326, 2004.
22. I.-J. Kim. Multi-window binarization of camera image for document recognition. In *Proceedings of International Workshop on Frontiers in Handwriting Recognition*, pages 323–327, 2004.
23. L. Vincent. Exact euclidean distance function by chain propagations. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 520–525, jun 1991.
24. J.-M. Chassery and A. Montanvert. Geometrical representation of shapes and objects for visual perception. In *Geometric Reasoning for Perception and Action*, volume 708 of *LNCS*, pages 163–182. Springer, 1993.
25. M.B. Dillencourt, H. Samet, and M. Tamminen. A general approach to connected-component labeling for arbitrary image representations. *Journal of the ACM*, 39(2): 253–280, April 1992.
26. G.M. Morton. A computer oriented geodetic data base; and a new technique in file sequencing. Technical report, IBM Company, 1966.
27. A. Antonacopoulos, S. Pletschacher, D. Bridson, and C. Papadopoulos. ICDAR 2009 page segmentation competition. In *Proc. of International Conference on Document Analysis and Recognition*, pages 1370–1374, 2009.
28. J. Fabrizio, B. Marcotegui, and M. Cord. Text segmentation in natural scenes using toggle-mapping. In *Proc. of IEEE Intl. Conference on Image Processing*, 2009.
29. B. Su, S. Lu, and C.L. Tan. Binarization of historical document images using the local maximum and minimum. In *Proceedings of the IAPR International Workshop on Document Analysis Systems*, pages 159–166, 2010.
30. J. Serra. Toggle mappings. Technical report, CMM, Ecole des Mines, France, 1989.
31. A. Antonacopoulos, D. Bridson, C. Papadopoulos, and S. Pletschacher. A realistic dataset for performance evaluation of document layout analysis. In *Proceedings of International Conference on Document Analysis and Recognition*, pages 296–300, july 2009.
32. I. Pratikakis, B. Gatos, and K. Ntirogiannis. ICFHR 2012 competition on handwritten document image binarization. In *Proc. of International Conference on Frontiers in Handwriting Recognition*, pages 813–818, 2102.
33. A.F. Mollah, S. Basu, and M. Nasipuri. Computationally efficient implementation of convolution-based locally adaptive binarization techniques. In *Wireless Networks and Computational Intelligence*, volume 292 of *CCIS*, pages 159–168. Springer, 2012.
34. R. Smith. An overview of the tesseract OCR engine. In *Proceedings of International Conference on Document Analysis and Recognition*, volume 2, pages 629–633, 2007.
35. P. Vandewalle, J. Kovacevic, and M. Vetterli. Reproducible research in signal processing. *IEEE Signal Processing Magazine*, 26(3):37–47, 2009.
36. G. Lazzara, R. Levillain, T. Géraud, Y. Jacquelet, J. Marquegnies, and A. Crépin-Leblond. The SCRIBO module of the Olena platform: a free software framework for document image analysis. In *Proc. of the Intl. Conf. on Document Analysis and Recognition*, 2011.
37. R. Levillain, T. Géraud, and L. Najman. Milena: Write generic morphological algorithms once, run on many kinds of images. In *Mathematical Morphology and Its Application to Signal and Image Processing (Proc. of the Intl. Symp. on Mathematical Morphology)*, pages 295–306. Springer, 2009.
38. R. Levillain, T. Géraud, and L. Najman. Why and how to design a generic and efficient image processing framework: The case of the Milena library. In *Proceedings of the IEEE International Conference on Image Processing*, pages 1941–1944, 2010.

Guillaume Lazzara completed his MSc at the School for Computer Science and Advanced Techniques (EPITA), France, in 2008. His research areas are image processing, document analysis and software engineering. He is currently working at EPITA Research and Development Laboratory (LRDE), Paris, France.

Thierry Géraud received a Ph.D. degree in signal and image processing from Télécom ParisTech in 1997, and the Habilitation à Diriger les Recherches from Université Paris-Est in 2012. He is one of the main authors of the Olena platform, dedicated to image processing and available as free software under the GPL licence. His research interests include image processing, pattern recognition, software engineering, and object-oriented scientific computing. He is currently working at EPITA Research and Development Laboratory (LRDE), Paris, France.