Constant-Time Locally Optimal Adaptive Binarization

Iuliu Konya Christoph Seibert Stefan Eickeler Sebastian Glahn
Fraunhofer Institute for Intelligent Analysis and Information Systems (IAIS)
Schloss Birlinghoven, 53754 Sankt Augustin, Germany
E-mail: {first-name.last-name}@iais.fraunhofer.de

Abstract

Scanned document images are nowadays becoming available in increasingly higher resolutions. Meanwhile, the variations in image quality within typical document collections increase due to images coming from different scan service providers, time periods or digitization methods. Binarization is a crucial first step for many document analysis algorithms. Adaptive thresholding algorithms have been shown to perform well on degraded documents, however their speed is orders of magnitude slower than that of global algorithms and they generally require manual fine-tuning of parameters for producing good results. This paper proposes a generic constant-time adaptive binarization algorithm, along with a constant-time method for automatically determining good window sizes for adaptive algorithms working on document images. Tests demonstrate a significant speedup compared to a straightforward implementation. Visual assessment of the results shows that the proposed method compares favorably with two well-known binarization techniques, and is especially suited for documents containing overexposed areas.

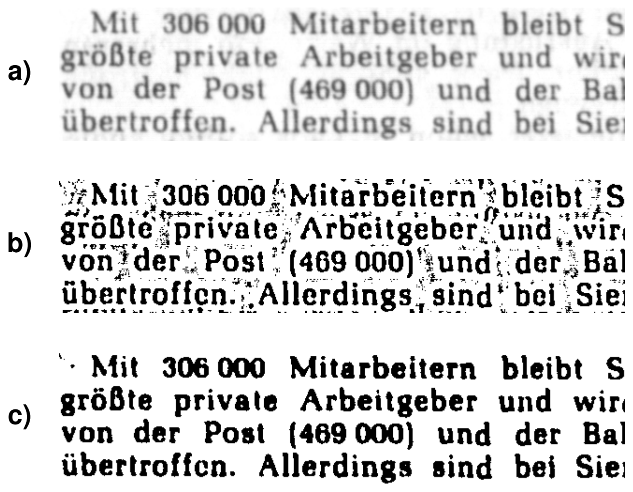
Keywords: *document binarization, local thresholding, adaptive thresholding, parameter estimation*

1. Introduction

Many of the current algorithms in document image analysis, be they concerned with skew estimation, geometric layout analysis, logical layout analysis or other issues, assume as input a binary image. In most cases however, input document images are either greyscale (with 256 grey levels) or color. The current state-of-the-art algorithms are able to perform the conversion from color to greyscale with minimal loss of relevant information even on very complex images [15]. In the current paper we assume having a greyscale image as input and will only be concerned with two-level thresholding, also known as binarization.

Approaches dealing with document image binarization can be divided into global or local. Global methods compute a single threshold value to classify the image pixels as object/foreground or background, whereas adaptive (local) methods use a different threshold value for each document area, selected according to certain local information. Arguably the most well-known global method was introduced by N. Otsu [9]. The good accuracy of the Otsu method was confirmed in [13], where it outperformed all other tested global thresholding methods. In a more recent survey [12] focused on artificial, text-only document images exhibiting constant illumination, the Otsu algorithm was again among the best performers. In general, global thresholding performs well for documents where there is a clear separation between foreground and background. Unfortunately, real-life document images frequently exhibit various kinds of degradations (e.g. illumination contrast, stains, noise) that weaken any guarantee for such a separation. Many adaptive binarization algorithms [4, 8, 11, 14] have been proposed in order to deal with these problems. Whereas the earlier binarization approaches were relatively straightforward, their complexity and run-time has risen continually and currently the trend is going towards multi-stage approaches [14] and combinations of binarization techniques [4]. The improved performance of the complex approaches comes at a price however: they generally depend on many (sometimes hidden) parameters, for which it is not trivial to find appropriate values so that the method performs well on a given document collection.

In this paper, we describe an adaptive binarization framework using at its core the technique of Otsu. The proposed algorithm runs in a time independent of the selected local window size. Furthermore, it is straightforward to implement, uses little additional memory and requires just two parameters, the sizes of the local windows. In order to show the genericity of the proposed method, the same framework was used to implement an adaptive version of the Kittler and Illingworth [6] global thresholding algorithm as the best performing technique in the recent comparison by Sezgin and Sankur [12]. Subsequently, we describe an automatic, constant-time method for estimating the required parameters from a given document image. An evaluation of the run-time performance of the proposed method against a standard implementation, as well as example result images are contained in section 4. Finally, conclusions are drawn in section 5.


a)
Mit 306 000 Mitarbeitern bleibt S
größte private Arbeitgeber und wir
von der Post (469 000) und der Bal
übertroffen. Allerdings sind bei Sie

b)
Mit 306 000 Mitarbeitern bleibt S
größte private Arbeitgeber und wir
von der Post (469 000) und der Bal
übertroffen. Allerdings sind bei Sie

c)
Mit 306 000 Mitarbeitern bleibt S
größte private Arbeitgeber und wir
von der Post (469 000) und der Bal
übertroffcn. Allerdings sind bei Sie

Figure 1. a) greyscale document image of size 605x142; b), c) its binarization using window size 11 and 33, respectively

2. Constant-time adaptive binarization

The basic idea for the proposed algorithm is straightforward: inspired by the success of Otsu's thresholding method on full document images, use the same technique for thresholding local windows. At his point it is worth noting that the global Otsu method computes an optimal threshold, i.e. one which maximizes a criterion related to the grey-level variance between objects and background, and thus all local thresholds will also be optimal. As input we consider to have the greyscale image and the radius r of the centered local window to use for each pixel. From the local histogram, one can then readily compute the optimal threshold according to Otsu's criterion in 2 passes over the histogram. Since the highest number of values investigated is equal to the histogram size (which is constant), it follows that this step can be performed in a time independent of the local window size r. Unfortunately, the Otsu thresholding method is based on the assumption that within the investigated window there actually exist two classes (i.e. both background and foreground). This does not hold for small, local windows, as they might completely be contained within the background or the foreground. Consequently, wrong thresholds and thus unwanted artifacts are produced by the algorithm for all such local windows. An example of this can be seen in figure 1. The problem can be solved by using a window size which is large enough to always contain both background and foreground. The results in this case are correct, but the use of large window sizes effectively cancels the "local" character of the method as well as the ability to properly adapt to a non-even illumination/contrast. Therefore, we propose using two centered, local windows—a small one for exploiting local information and a larger one which is guaranteed to always contain both background and foreground pixels. Thus, by applying Otsu's thresholding method on the weighted sum of the two local histograms one can determine an optimal threshold which makes use of both local and global information. In all our experiments, the weighting factor for the histogram of the small window was set to be equal to $Area(SmallWindow)/Area(LargeWindow)$, whereas the weight for the larger window histogram was set to 1.

In order to dramatically improve the run-time performance of the algorithm we build upon the method for performing median filtering in amortized constant time recently introduced in [10]. The median filtering algorithm needs to maintain one histogram for each column in the image. Each column histogram accumulates the vertical stripe of $2r+1$ pixels centered on the image row being processed. The algorithm also needs to maintain the histogram of the local window (i.e. the window centered on the pixel for which the threshold is to be determined). Initially, the local histogram can be computed by summing up the $2r+1$ column histograms centered on the start pixel. Afterwards, assuming the image traversal is done left-to-right, the local histogram can be computed for the pixel $i+1$ by subtracting the column histogram $i-r$ and adding the column histogram $i+1+r$. Note that the number of operations required by the addition and subtraction of histograms is independent of the local window size and is constant. When moving from one row to the next one under it, each of the column histograms is updated by subtracting the topmost pixel value inside it and adding the pixel value in the image directly below it. This is obviously also a constant time process. However, as proposed in the original algorithm, at the beginning of each row the local histogram needs to be initialized again with the sum of the nearby column histograms. This means that a penalty of $O(r)$ histogram additions is incurred for each row of the input image. The penalty can be reduced by alternating left-to-right and right-to-left scans of the image. In this way, at the end of each row the local histogram just needs to be moved one row down—all other pixels within it are unchanged. Updating of the local histogram can thus be accomplished with just r pixel additions and r pixel subtractions, a cost significantly lower than the r histogram additions otherwise necessary. A pseudocode description of the final version of the proposed algorithm (using two local windows) at the end of the current section.

We have performed experiments with a version of the above algorithm which uses a single local window of the same width as the document image (dubbed row-adaptive). For the selection of the height of the window one must also ensure that it is always tall enough to contain both background and object pixels. The necessary height can be computed automatically by a simple change in the second algorithm presented in section 3. Furthermore, using the same framework we have been able to transform the Kittler and Illingworth [6] minimum error global thresholding method into a constant-time adaptive algorithm. More details about this may be found in section 4.

Algorithm 1 Proposed binarization algorithm

```
Input: Image $X$ of size $m \times n$, local window radii $r_1 < r_2$
Output: Image $Y$ of the same size
  Initialize window histograms $Hs$, $Hl$ and column histograms $hs_{1 \dots n}$, $hl_{1 \dots n}$
  $K \leftarrow r_2^2 / r_1^2$
  for $i = 1$ to $m$ do
    if $i$ is odd then
      for $j = 1$ to $r_1$ do
        Remove $X_{i-r_1-1,j+r_1}$ from $Hs$
        Add $X_{i+r_1,j+r_1}$ to $Hs$
      end for
      for $j = 1$ to $r_2$ do
        Remove $X_{i-r_2-1,j+r_2}$ from $Hl$
        Add $X_{i+r_2,j+r_2}$ to $Hl$
      end for
      for $j = 1$ to $n$ do
        Remove $X_{i-r_1-1,j+r_1}$ from $hs_{j+r_1}$
        Add $X_{i+r_1,j+r_1}$ to $hs_{j+r_1}$
        Remove $X_{i-r_2-1,j+r_2}$ from $hl_{j+r_2}$
        Add $X_{i+r_2,j+r_2}$ to $hl_{j+r_2}$
        $Hs \leftarrow Hs + hs_{j+r_1} - hs_{j-r_1-1}$
        $Hl \leftarrow Hl + hl_{j+r_2} - hl_{j-r_2-1}$
        $Y_{i,j} \leftarrow OtsuThreshold(Hl + Hs \times K)$
      end for
    else
      the same, just traverse and update from right to left
    end if
  end for
```

3. Automatic window size estimation

This section contains the description of two distinct algorithms for automatically detecting the size of the local windows. Throughout this section we will assume that a preliminary binarized version of the input image is already available and that all its connected components have been determined. Note that the considered binary image need not have a good quality, and it is sufficient to create it by global binarization [9] or by employing the fast row-adaptive binarization method mentioned at the end of section 2. Many connected component labeling algorithms which run in $O(Numpix)$ time and have negligible additional memory requirements can be found in the literature, e.g. [2].

For estimating the size of the smaller local window we use the observation that local algorithms such as those proposed by Niblack [8] and Sauvola et al. [11] perform best when the local window covers about 1–2 characters. Note that in the specialized literature, there already exist methods for estimating the average character size of a given document image. Gatos et al. [3] estimate the average character height by sampling the image pixels randomly, following the contour of the containing connected component and finally computing an average of the components' heights.

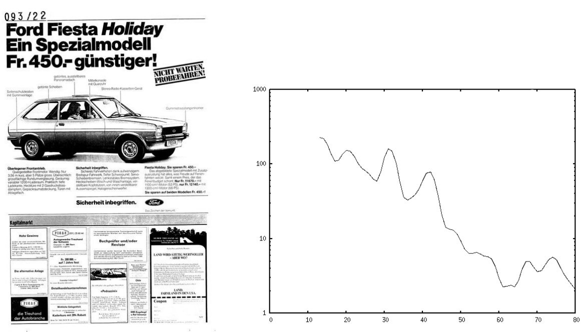


Figure 2. Document image with multiple font sizes and its corresponding smoothed histogram of connected component heights

Zheng et al. [16] compute the average character width and height in a document by taking the highest peak in the histogram of connected component widths and heights, respectively. The drawback of these methods is that they generally fail on documents which exhibit one or more of the following characteristics: the page contains less text than halftones or drawings, there exist multiple font sizes and there exists significant noise in the image.

Our method attempts to partially cope with such situations. In order for the local binarization to perform best on text, it is advantageous to use as reference for the dimensions of the window the character size of the dominant font (i.e. the font in which the most characters are typed), not simply the average character size on the page. For pages containing multiple font sizes, the average character size will most likely be different from that of the dominant font, thus using it will lead to sub-optimal binarization results for the dominant font. Another useful observation is that for printed texts, the character height for a specific font size fluctuates much less than its character width, and thus it can generally be estimated more robustly. Finally, since in the majority of cases the scanning resolution of the document is known to the binarization algorithm, one may optionally ignore all connected components with heights smaller than a certain value V. In our experiments, we have used $V = 0.09cm$, motivated by the fact that printed characters smaller than this are generally not legible by humans. In case the scanning resolution is unknown, the value V can be set to e.g. 5 pixels, as any text smaller than this will not be legible either by humans or by optical character recognition systems. As input data, we take a list containing the bounding boxes of all connected components on the document page and optionally, the scanning resolution used. The algorithm operates in two passes: first, a smoothed histogram is created from the heights of all given connected components. Subsequently, a first approximation of the dominant height is computed by searching for a high and flat peak starting from the value V (in pixels). The peak is determined by going through the histogram in ascending order and searching for the (first) value i where the following two conditions are met:

- $H_i \geq H_{i-1}$ and $H_i \geq H_{i+1}$
- $\frac{min(log(H_i - H_{i-1}+i+1), log(H_i - H_{i+1}+i+1))+1}{H_{i-1}+H_i+H_{i+1}+1}$ is lowest

After finding the value i_H, a new smoothed histogram is created by using only the connected components having their height greater or equal to i_H. In the second run, a new peak having the property described above is searched in the histogram, and the obtained height is saved as the desired result. We have experimentally determined that the above method provides a robust estimate for the dominant character height in the majority of cases when there exist at least 1–2 lines of text on the page (regardless of the presence of other non-text content).

The second algorithm of this section computes an estimate for the size of the larger local windows used in the proposed binarization method. As mentioned in section 2, the larger local window needs to always contain both background and foreground pixels in order for the Otsu thresholding method to work properly. At this point, it is interesting to note that Huang et al. [5] recently proposed the automatic estimation of the local window size based on the Lorentz information measure. This is unfortunately not applicable in our situation, since we need a hard guarantee for the existence of both types of pixels within the window. For reaching our goal, we employ a constant-time approximate Euclidean distance transform (EDT) algorithm [1] on the inverted binary image from which all connected components with height lower than the small window size have been erased. Afterwards, the size of the larger local window is set to be equal to one plus the maximum value in the distance transformed image. Note that there exist exact EDT algorithms running in $O(Numpix)$ time [7], however their hidden constant is generally much higher than the one in approximate algorithms.

4. Experiments

We have tested the runtime speed of the proposed method against that of a straightforward implementation of the same algorithm on a typical document image of size 2336x3503 (approx. DIN-A4 scanned at 300dpi). All experiments were performed on a computer equipped with a Core2Duo 2 GHz processor, and for each windows size the algorithms were run 5 times (the average value was plotted). Note that we have tested the run-time performance of the single-window approach, not that of the 2-window method. This is due to the fact that the relative performance would likely remain the same (2 histograms would have to be updated instead of a single one) and because we wanted to include in the graph the row-adaptive approach, which only uses a single window size. From the plot one can clearly see the performance gain growing proportionally to the window size. The speed of the row-adaptive binarization is extremely high and directly comparable to that of a global algorithm. Whereas for a window of size 11 both other algorithms take about 23 seconds, for a window of size 251 the proposed algorithm performs more than 30x faster than the regular implementation. On the same image, the average time for automatically computing the optimal window sizes was under 1 second, whereas the proposed fully automatic 2-window approach took about 48 seconds.

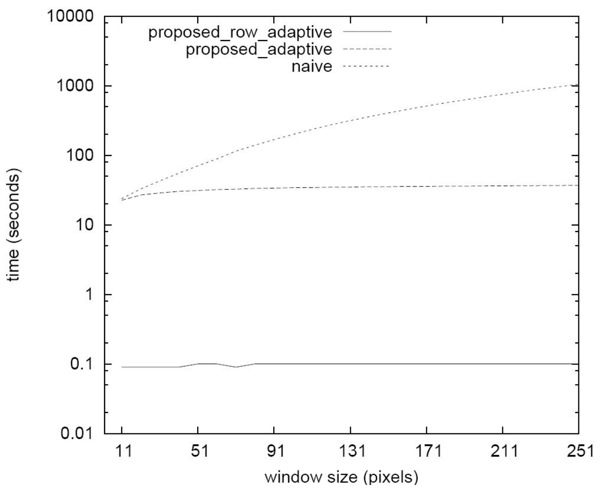


Figure 3. Timing comparison for proposed method. Note the log-scale applied to the Y-axis

An adaptive variant of the Kittler et al. [6] minimum error global thresholding was also implemented using the described fully automatic framework and had its runtime performance tested. It was found that it is about 6 times slower than the adaptive method using Otsu thresholding. A simple, yet effective way of significantly reducing the runtime of both methods would be to use a regular grid to sample the input image, compute the thresholds just for the grid nodes and finally use interpolation (e.g. bilinear) for computing the rest of the thresholds. The size of the grid cells can be experimentally derived from the already computed dominant character height.

From figure 4 one can see that the 2-window approach with automatically determined parameters produces very good results in comparison with the global binarization algorithm of Otsu and the Sauvola et al. algorithm. For the Sauvola et. al algorithm we have also used the automatically computed small window size (other values perform worse). The poor performance of the Sauvola et al. algorithm is due to their implicit assumption that text is nearly black, which is false in case of overexposed areas. We have also investigated the results on stained and badly degraded images where on certain areas the background and text values were also very close to each other, but had much lower grey values. In this situation, the Sauvola et al. algorithm produced better results than the proposed algorithm, which in turn performed better than global binarization. Note that the Sauvola et al. method works generally well on text, but has the undesirable effect of completely destroying halftones and hollowing out characters much larger than the window size. Our algorithm does not exhibit such prob-

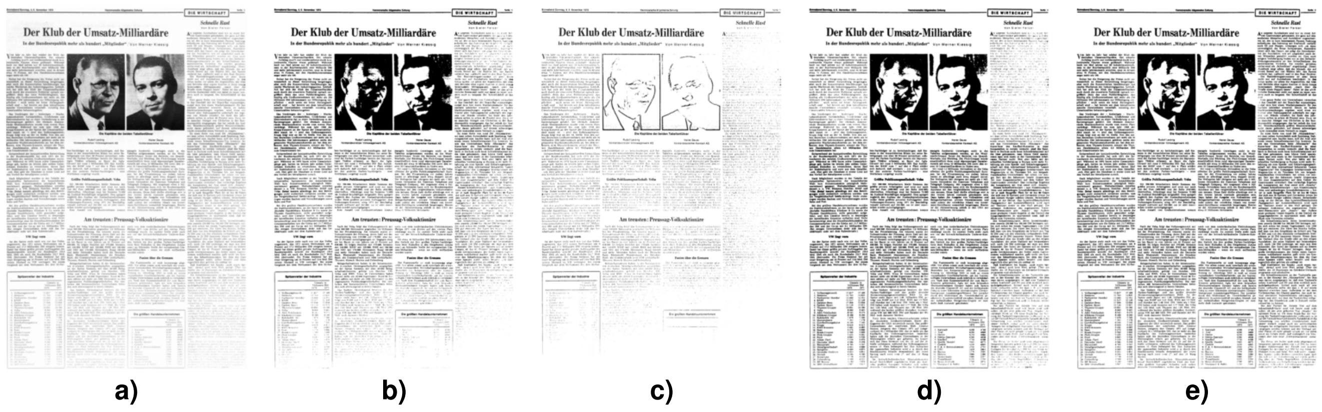


Figure 4. a) greyscale document image and its binarized versions using b) Otsu global thresholding; c) Sauvola et al. method; d) proposed row-adaptive method and e) proposed adaptive method with automatically determined window sizes

lems, due to the mixture of local and global information utilized for selecting the thresholds. It is also worth noting that while the row-adaptive method is certainly less versatile than the proposed baseline method, it is much faster and works equally well on documents for which the contrast changes occur from top to bottom (i.e. the light source was directly above or below the document image).

5. Conclusions

In this paper, a novel two-window approach for adaptive binarization was proposed, which makes use of both local and global information. It has the desirable property that its run time is independent of the local window size. We also proposed methods for computing the optimal values for the required parameters. Both the presented binarization framework and the parameter estimation methods are applicable to other binarization techniques, as demonstrated by our adaptation of the Kittler et al. algorithm. The method was shown to produce very good results on documents containing overexposed areas. Future research will focus on detecting the most suitable binarization algorithm for a given document area.

Acknowledgement

This work was supported by the German Federal Ministry of Economics and Technology (BMWi) funded program *Theseus* in the project *Contentus*.

References

[1] P. E. Danielsson. Euclidean distance mapping. *Computer Graphics and Image Processing*, 14:227–248, 1980.

[2] M. B. Dillencourt, H. Samet, and M. Tamminen. A general approach to connected-component labeling for arbitrary image representations. *J. ACM*, 39(2):253–280, 1992.

[3] B. Gatos, D. Danatsas, I. Pratikakis, and S. J. Perantonis. Automatic table detection in document images. In *Proc. 3rd Int. Conf. on Advances in Pattern Recognition (ICAPR05), LNCS 3686*, pages 609–618, 2005.

[4] B. Gatos, I. Pratikakis, and S. J. Perantonis. Improved document image binarization by using a combination of multiple binarization techniques and adapted edge information. In *Proc. 19th Int'l Conf. Pattern Recognition*, pages 239–244. IEEE Computer Society, 2008.

[5] Q. Huang, W. Gao, and W. Cai. Thresholding technique with adaptive window selection for uneven lighting image. *Pattern Recogn. Lett.*, 26(6):801–808, 2005.

[6] J. Kittler and J. Illingworth. Minimum error thresholding. *Pattern Recognition*, 19:41–47, 1986.

[7] C. R. Maurer, R. Qi, and V. Raghavan. A linear time algorithm for computing exact euclidean distance transforms of binary images in arbitrary dimensions. *Pattern Analysis and Machine Intelligence*, 25(2), 2003.

[8] W. Niblack. *An Introduction to Digital Image Processing*. Prentice-Hall, Englewood Cliffs, NJ, 1986.

[9] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Trans. on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

[10] S. Perreault and P. Hebert. Median filtering in constant time. *IEEE Trans. on Image Processing*, 16(9):2389–2394, 2007.

[11] J. Sauvola and M. Pietikainen. Adaptive document image binarization. *Pattern Recognition*, 33:225–236, 2000.

[12] M. Sezgin and B. Sankur. Survey over image thresholding techniques and quantitative performance evaluation. *J. Electronic Imaging*, 13(1):146–165, January 2004.

[13] O. D. Trier and A. K. Jain. Goal-directed evaluation of binarization methods. *Pattern Analysis and Machine Intelligence, IEEE Trans.* , 17(12):1191–1201, Dec 1995.

[14] S. Wu and A. Amin. Automatic thresholding of gray-level using multi-stage approach. In *Proc. ICDAR*, pages 493–497. IEEE Computer Society, 2003.

[15] J. Yi, Y. Peng, and J. Xiao. Color-based clustering for text detection and extraction in image. In *Proc. Int. Conf. on Multimedia*, pages 847–850, New York, USA, 2007. ACM.

[16] Y. Zheng, C. Liu, X. Ding, and S. Pan. Form frame line detection with directional single-connected chain. In *Proc. ICDAR*, pages 699–703. IEEE Computer Society, 2001.