Using Scale-Space Anisotropic Smoothing for Text Line Extraction in Historical Documents

Rafi Cohen[1](✉), Itshak Dinstein[2], Jihad El-Sana[1], and Klara Kedem[1]

[1] Department of Computer Science, Ben-Gurion University, Beer-Sheva, Israel
{rafico,el-sana,klara}@cs.bgu.ac.il
[2] Department of Electrical and Computer Engineering,
Ben-Gurion University, Beer-Sheva, Israel
dinstein@ee.bgu.ac.il

Abstract. Text line extraction is vital pre-requisite for various document processing tasks. This paper presents a novel approach for text line extraction which is based on Gaussian scale space and dedicated binarization that utilize the inherent structure of smoothed text document images. It enhances the text lines in the image using multi-scale anisotropic second derivative of Gaussian filter bank at the average height of the text line. It then applies a binarization, which is based on component-tree and is tailored towards line extraction. The final stage of the algorithm is based on an energy minimization framework for removing spurious text line and assigning connected components to lines. We have tested our approach on various datasets written in different languages at range of image quality and received high detection rates, which outperform state-of-the-art algorithms. Our MATLAB code is publicly available. (http://www.cs.bgu.ac.il/~rafico/LineExtraction.zip)

Keywords: Historical document processing · Text lines extraction

1 Introduction

Many of the document analysis algorithms, such as indexing, word retrieval and text recognition, expect extracted text lines, as an input. Thus, text line extraction is an essential operation in document processing and a substantial number of related algorithms have been published. Most of these algorithms expect binary images and some are designed to handle gray scale images.

Smearing based methods [4,10,15] apply Gaussian based filtering and binarization to enhance line structure. These approaches yield good results and became popular methods for text line extraction (ranked 1st in ICDAR 2009 and ICFHR 2010 contests [8,9], and 3rd in ICDAR 2013 contest [16]). However, the performance of these methods depends on choosing the correct scale of the Gaussian based filter. Most authors do not provide an algorithm for choosing the correct scale [10,15] or choose the scale based on ad-hoc heuristics [4]. The binarization phase also inherits the limitations of the adapted binarization algorithm which is either ad-hoc binarization [15] or based on active-contours [4,10]

© Springer International Publishing Switzerland 2014
A. Campilho and M. Kamel (Eds.): ICIAR 2014, Part I, LNCS 8814, pp. 349–358, 2014.
DOI: 10.1007/978-3-319-11758-4_38

which are computationally slow. Seam-based line extraction algorithm compute an energy map, which is used to guide the progress of the seam that determine the text lines or their boundaries. The algorithm is required to determine the boundary seams of the detected text-lines, which is done using ad-hoc heuristics [14].

In this paper we present a novel method designed to detect text lines. Our algorithm is based on robust theoretical background, i.e., scale space theory [11] and our binarization method is fast, and tailored towards line extraction in documents. In an initial step our approach enhances the text lines in the image using multi-scale anisotropic second derivative of Gaussian filter bank at the average height of the text line. It then applies a binarization, which is based on component-tree and utilizes the structure of smoothed text line.

In the rest of the paper we overview closely related work and background literature. We then present our algorithm and its experimental evaluation. Finally we conclude and draw directions for future work.

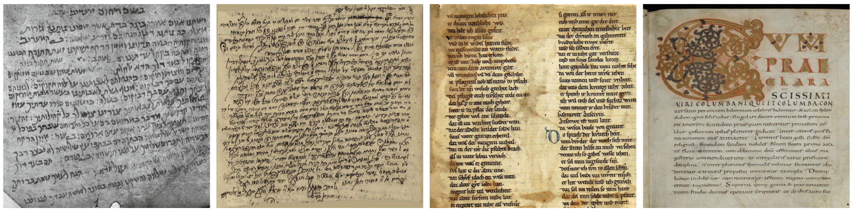
2 Related Work

Text line extraction algorithms could be categorized into projection-based methods [2], grouping methods [7,13], seam-based algorithm [14] and smearing methods [4,10,15].

Projection-based algorithms divide the document image into vertical strips and horizontal projections are calculated within the stripes. The resulting projections are combined in order to extract the final text lines. Bar-Yosef *et al.* [2] applied an oriented local projection profiles (LPP) inside a sliding stripe. The average skew of the current stripe is calculated and the next stripe is projected along that skew direction. Grouping methods extract text lines by aggregating units in a bottom-up strategy. The units may be pixel or higher level representation, such as connected components, blocks or other features such as interest points. Rabaev *et al.* [13] used a sweep-line to aggregate connected components, that correspond to characters, into text lines. A seam-carving-based approach has been developed recently. Saabni *et al.* [14] used two types of seams, medial and separating. Both types of seems propagate according to energy maps, which are defined based on the distance transform of the gray scale image. The seams tend to diverge when big gaps between words or holes in the document are present.

Smearing approaches enhance line structure and then apply binarization to extract text lines. Shi *et al.* [15] converted an input image into an adaptive local connectivity map (ALCM), where the value of each pixel is defined as the cumulative intensity of the pixel inside a window of a predefined size. Finally the ALCM image is binarized to extract text line patterns. The method do not contain a mechanism for determining the appropriate scale of the filter for degraded grayscale historical documents and the binarization algorithm is not tailored towards lines extraction. A popular variant of the smearing method [4,10] is based upon convolving the image with an anisotropic Gaussian (or a bank or Gaussians) followed by segmentation of text lines using active contours. Bukhari *et al.* [4] suggest to choose the scales of the Gaussians by binarizing the document and inspecting its height histogram, which is susceptible to noise in degraded documents, see Fig. 1(a). Another drawback for the level-set based active contours methods [10] is their complex and slow computation.

Despite considerable progress over the last decade, automatic text line segmentation of historical documents, as those presented in Fig. 1, remains an open problem.



(a) (b) (c) (d)

Fig. 1. Samples of the documents on which we perform our tests. (a) Genizah handwritten manuscript; (b) Pinkasim handwritten cursive manuscript; (c) German manuscript from Parzival dataset; (d) Latin manuscript from Saint Gall dataset.

3 Notations and Definitions

Our approach relies on scale space scheme and utilize component tree to extract text lines. To simplify the presentation of our algorithm we briefly overview these two topics.

3.1 Scale-Space Overview

Scale space can be intuitively thought of as a collection of smoothed versions of the original image. Formally, given an image $I : \mathbf{R^2} \rightarrow \mathbf{R}$, its linear scale-space representation $L : \mathbf{R^2} \times \mathbf{R^2_+} \rightarrow \mathbf{R}$ can be defined by convolution with anisotropic Gaussian kernels of various lengths $\sqrt{t_x}$ and $\sqrt{t_y}$ in the coordinate directions, defined as $L(x,y;t_x,t_y) = g(x,y;t_x,t_y) * I(x,y)$, where $g : \mathbf{R^2} \times \mathbf{R^2_+} \rightarrow \mathbf{R}$ is an anisotropic Gaussian defined in Eq. 1. We define a multiplication factor η as $\frac{\sigma_x}{\sigma_y}$, where σ_i is related to t_i by $\sigma_i = \sqrt{t_i}$.

$$g(x,y;t_x,t_y) = \frac{1}{2\pi\sqrt{t_x t_y}} e^{-\left(\frac{x^2}{2t_x} + \frac{y^2}{2t_y}\right)}. \quad (1)$$

We denote by $\partial_{x^\alpha} L(x,y;t_x,t_y)$ the partial derivative of L with respect to x, where L is differentiated α times. Lindeberg [11] showed that the amplitude of spatial derivatives, $\partial_{x^\alpha}\partial_{y^\beta} L(x,y;t_x,t_y)$, in general *decrease with scale*, i.e., if an image is subject to scale-space smoothing, then the numerical values of spatial derivatives computed from the smoothed data can be expected to decrease.

If two signals f and f' are related by scale, i.e., $f(x) = f'(sx)$, then it is possible to normalize that spatial derivative of the scale-space such that the normalized derivatives are equal [11]. More formally, Let the scale space representation of f and f' be given as $L(x;t) = g(x,t) * f$ and $L'(x';t') = g(x',t') * f'$, where the spatial variables and the scale parameters are transformed according $x' = sx$ and $t' = s^2 t$. Then, if γ-normalized function of the derivatives is defined as $\partial_\xi = \sqrt{t}\partial_x$ and $\partial'_\xi = \sqrt{t'}\partial_{x'}$ then $\partial_{\xi^\alpha} L(x;t) = \partial_{\xi'^\alpha} L'(x';t')$. That is, the γ-normalized function of the derivatives are *scale invariant*.

3.2 Component-Tree

The level sets of a map are the sets of points with level above a given threshold. The inclusion relation enables connected components of the level sets to be organized in a tree structure, which is called the *component tree* [12]. We denote the threshold set obtained by thresholding a map with threshold t by $\mathcal{B}_t$ and the set of connected components in $\mathcal{B}_t$ by $\mathcal{C}_t$. The nodes in a *component-tree* correspond to the components in $\mathcal{C}_t$ for varying values of the threshold t. The root of the tree is the member of $\mathcal{C}_{t_{\min}}$, where $t_{\min}$ is chosen such that $|\mathcal{C}_{t_{\min}}|$=1. The next level in the tree correspond to $\mathcal{C}_{t_{\min}+d}$, and in general the nodes in the tree that belong to level ℓ correspond to $\mathcal{C}_{t_{\min}+\ell d}$, where d is a parameter that determines the step size for the tree. There is an edge between $C_i \in \mathcal{C}_t$ and $C_j \in \mathcal{C}_{t+1}$ if and only if $C_j \subseteq C_i$. The maximal threshold $t_{\max}$ used in tree construction is simply the maximal value in the map.

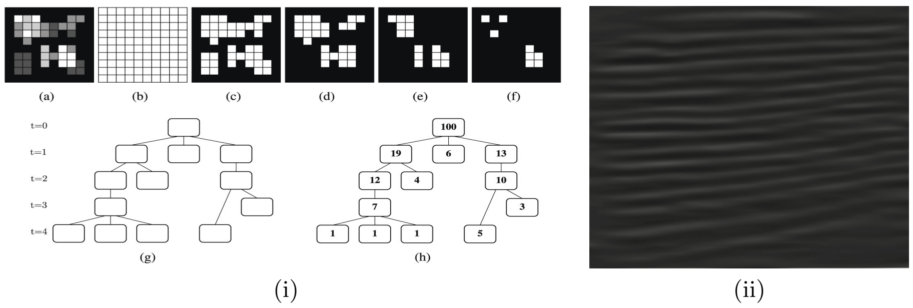


Fig. 2. (i)(a) A gray-level image F and its successive threshold sets $B_t(F)$ for t from 0 (b) to 4 (f), where $d = 1$; (g) The component-tree of F. (h) The same tree, enriched by an attribute (the size of the connected component of each node), courtesy of [12]. (ii) Lines enhancement result.

4 Our Approach

In this paper we describe a text line segmentation approach for handwritten documents, which is based on Gaussian scale space and component-tree traversal.

The method starts by enhancing lines in the image, using multi-scale anisotropic second derivative of Gaussian filter bank. The resulting image is binarized using component-tree traversal that is tailored towards line extraction. At the final step, spurious detected lines that do not correspond to text lines are removed.

4.1 Lines Enhancement

The pixels in an image can be regarded as two dimensional random variables. They are generated by an unknown Probability Distribution Function (PDF), which represents the distribution of text lines. Specifically, the PDF is continuous and has smaller values (dark) in the text line area, while there are larger values (bright) in the gap and marginal area [10]. Valleys on the probability map represent text lines, while peaks are the boundary between neighboring text lines. As a result of this structure, a convolution of text line with a second derivative of an anisotropic Gaussian, elongated along the horizontal direction generates ridges along text lines and valleys along the gaps between text lines [5]. Making it an appropriate filter for enhancing the lines structure in a document.

The Appropriate scale for this filter correspond to the text line height, which varies along the text line itself, due to ascenders and descenders, and along different text lines. We use a multi-scale filtering and detect the optimal scale for each point using the scale-space framework [11]. We construct a scale space representation of the images by convolving the image with the γ-normalized function of g_{xx} from Eq. 1 with $\eta > 1$, and choosing for each pixel the strongest response along the scale-space. The scales at which the image is convolved with corresponds to the height range of the characters in the document. A robust estimate of character height range in gray-scale images is obtained using the evolution map (EM) tool introduced by Biller *et al.* [3]. The EM supplies details about the height range of the characters in the document, without binarization. For binary images the range is taken as $(\mu, \mu + \sigma/2)$, where μ and σ are the average and standard deviation of the heights of the connected components in the document. Fig. 2(ii) illustrates the result of lines enhancement on a document from Fig. 1(a).

4.2 Text Line Extraction Using Component-Tree

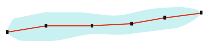
To extract the text lines we need to binarize the gray scale image, $\mathcal{R}$, resulting from applying the Gaussian scale space on the original image (Section 4.1). Off-the-shelf general binarization algorithm do not take into account the properties of the resulting image, require tuning parameters, and often introduce noise and artifacts. Instead, we apply a binarization procedure, which is based on *component-tree* scheme and geared toward the structure of $\mathcal{R}$.

A connected component that represents a text line resembles a thick simple polyline that covers the entire text line (the thickness is not uniform). Motivated by this observation, we build a component-tree of $\mathcal{R}$ and for each connected component, C_i (represented by the node, $node(C_i)$, in the tree) we measure how well C_i can be represented by a simple piecewise linear approximation. Let us refer to this measure as $F(C_i)$. One could compute $F(C_i)$ by detecting the skeleton of the component and measuring its linearity, but skeleton structure is not robust and sensitive to noise. Instead, we fit a simple piecewise linear approximation from the left end to the right end of the component and measure quality of the fit. We chose to implement that by fitting a uniform least squares spline of order 1 with k knots for the connected component, C_i. The first and last points correspond to the left and right end of the component and the $k-2$ remaining points are distributed uniformly along the line connecting the two end points, as depicted in Fig. 3(b). The fit quality is computed based on two terms: (a) the average distance of each pixel from the spline, and (b) the difference between the area of the component and the sum of the distances of the contour pixels from the spline. The average distance is compared with the average letter height to detect and refine component that include two consecutive lines. The second term is used to detect partial merge of adjacent text lines that form a non-convex component.

To extract the text lines we traverse the component-tree top-down and at each node, $node(C_i)$ we measure its fitness, $F(C_i)$, and based on that we determine whether it represents a text line or not. If C_i represent a text line we output this text line and the search along this branch is complete, otherwise we refine the component by recursively processing the children of the $node(C_i)$. Fig. 3(a) presents the pseudo-code of the traversal procedure.

```
 1: $Ouput = \phi$.
 2: Enqueue the root node $v$ into a queue $Q$
 3: **while** $Q$ is not empty **do**
 4:     $C_i \leftarrow Q$.dequeue()
 5:     **if** $F(C_i)$ represents a text line **then**
 6:         $Ouput = Ouput \bigcup C_i$.
 7:     **else**
 8:         Enqueue all children of $C_i$ into $Q$.
 9:     **end if**
10: **end while**
11: return $Output$.
```

(a)



(b)

Fig. 3. (a) Traversal Algorithm; (b) a synthetic blob with an approximating spline (in red) that uses 6 knots (k=6)

4.3 Post-Processing

Our algorithm usually extracts the correct text line efficiently. However, in some cases it includes spurious lines that do not correspond to text lines or split a text lines into disconnected segments, as shown in Fig. 4(c). We overcome these limitations by detecting and removing spurious text lines and connecting segments that belong to the same text line. This stage involves minimizing an energy function on a binarized version of the document.

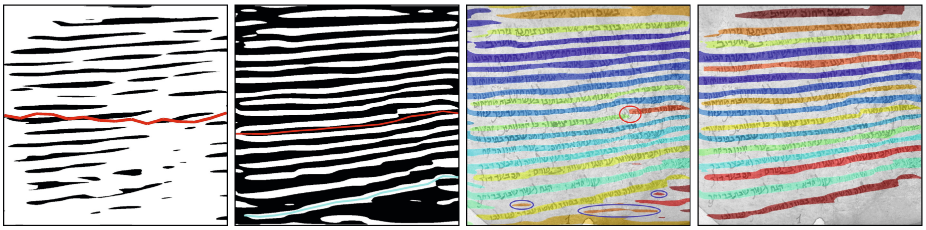


Fig. 4. The various stages of the component tree traversal, where splines used to evaluate piecewise linearity are depicted in red (unsuccessful fit) or cyan (successful fit), (a) the root of the component tree, at $\mathcal{C}_{-17}$, (b) the components at $\mathcal{C}_{13}$, (c) the result before discarding spurious lines (disconnected segments are encircled in red, and some spurious lines are encircled in blue) and (d) the final result.

Our approach relies on multi-label graph cut minimization [6] where graph cuts are used to approximate energy minimization of arbitrary functions. Let $\mathcal{L}$ be the set of lines (labels and lines are interchangeable throughout this section) that were extracted in Section 4.2 (Fig. 4(b)) and let $\mathcal{C}$ be the set of connected component in the document. The goal is to find a labeling f that assigns each component $c \in \mathcal{C}$ a label $\ell_c \in \mathcal{L}$, where f is consistent with the observed data, spatial coherent and uses a minimal set of labels (i.e., lines). The energy function, $E(f)$ defined in Eq. 2, consists of three terms: the data cost, the smoothness terms and the label cost. Minimizing the energy function, $E(f)$, produces an appropriate labeling.

$$E(f) = \sum_{c \in \mathcal{C}} D(c, \ell_c) + \sum_{\{c,c'\} \in \mathcal{N}} d(c, c') \cdot \delta(\ell_c \neq \ell_{c'}) + \sum_{\ell \in \mathcal{L}} h_\ell \cdot \delta_\ell(f) \tag{2}$$

The cost term, $D(c, \ell_c)$, expresses the cost of assigning c the label ℓ_c and is defined as the Euclidean distance between the centroid of c and the line represented by ℓ_c. The smoothness term determines the coherence of the labels ℓ_c and $\ell_{c'}$ with the spatial relation of the components c and c'. That is, the closer the components are the higher is the chance that they got assigned the same label. Let $\mathcal{N}$ be the set of adjacent component pairs. We set $|\mathcal{N}| = 2$ and define the distance $d(c, c')$ in Eq. 2 according to $d(c, c') = \exp(-\alpha \cdot d_e(c, c'))$ (the spatial coherence strength decays exponentially with Euclidean distance). The term $d_e(c, c')$ is the Euclidean distance between the centroids of components c and c', and the constant α is defined as $(2 \langle d_e(c, c') \rangle)^{-1}$, where $\langle \cdot \rangle$ denotes expectation over all pairs of adjacent elements [5]. The term $\delta(\ell_c \neq \ell_{c'})$ is Kronecker's delta. The label costs penalize each unique label that appears in f, where h_ℓ is the non-negative label cost of label ℓ, and $\delta_\ell(f)$ is an indicator function that is assigned 1, when the label ℓ appears in f and 0 otherwise. We define the *density* of a line ℓ as the number of foreground pixels in the binarized document overlapping with ℓ, and r_ℓ as the the ratio between the density of ℓ and the maximal density in $\mathcal{L}$. The label cost h_ℓ is defined as $\exp(-\beta \cdot r_\ell)$, where β is a constant we set experimentally.

Finally, we merge broken line segments. For each segment we extract its left and right endpoints and define the direction of a component as the vector connecting the left endpoint to the right endpoint. Two adjacent component are merged if (a) the direction of the vector connecting the two components (the right of the first component to the left of the second one) falls between the direction of the two components and (b) their vertical distance is less than the average letter height.

5 Experimental Results and Discussions

We evaluated our text line detection on various datasets and received encouraging results. The test datasets include documents written by different writers and in various languages. Hence, the presented methodology is script and writer independent and copes nicely with noise. the datasets are ICDAR 2013 [16], ICDAR 2009 [8], Hebrew [13], Saint Gall [7] and Parzival [1] datasets. ICDAR 2013 contains 150 pages written in English, Greek and Bangla. ICDAR 2009 contains 200 pages written in English, French, German and Greek. The Hebrew dataset contains 58 degraded pages from Cairo Genizah collection and 6 pages from the Pinkasim collection. The Saint Gall database contains 60 pages of a Latin manuscript from the 9th century. The Parzival includes 47 pages of a German manuscript from the 13th century. For Parzival we used the ground-truth generated by [13].

The performance evaluation is based on a MatchScore [16] that computes the maximum overlap of a text region with the ground truth region. If this score is above a given threshold T_α, the text line is considered as correct (one-to-one match, o2o). Based on this MatchScore, the Detection Rate (DR), the Recognition Accuracy (RA) and the Performance Metric (FM) are defined using Eq. 3, where N and M are the number of text lines in the ground truth and the number of text lines detected by the algorithm, respectively. In our experiments we set T_α as 95% for datasets of binary images, and 90% for datasets of gray-scale images. For all datasets and all algorithms the performance evaluation is based on a binarized version of the datasets. For Saint Gall and Parzival we measure the performance by means of the Pixel-Level Hit Rate (PHR) and the FM (also called Line Accuracy Measure) as in [1,7]. The results of the presented algorithm are reported in Table 1, we also mention for each dataset whether it consists of binary pages (B) or gray-scale pages (G). Throughout the experiments we have used the 15 knots (k=15) to measure the linearity of the components, the scale space is defined based on $d = 1$ and $\eta = 3$.

$$DR = \frac{o2o}{N}, RA = \frac{o2o}{M}, FM = \frac{2 \times DR \times RA}{DR + RA} \tag{3}$$

Although the algorithm achieves high detection rates, it suffers some limitations. For example, if salient objects in the image, such as holes and drawings,

Table 1. Results on different datasets compared with known state-of-the-art algorithms. Each dataset contains either binary (B) or gray-scale documents (G).

	Our Method					state-of-the-art	
	M	o2o	DR	RA	FM	FM	
ICDAR 2013 [16](B)	2651	2621	98.94%	98.86%	**98.90%**	98.66%	
ICDAR 2009 [8](B)	4033	4021	99.67%	99.70%	**99.69%**	99.53%	
Hebrew [13](G)	1257	1154	89.04%	91.88%	**90.44%**	86.10%	
	PHR			FM		PHR	FM
Saint Gall [7](G)	**99.08%**			**99.22%**		98.94%	99.03%
Parzival [1](G)	**98.31%**			**97.88%**		96.30%	96.40%

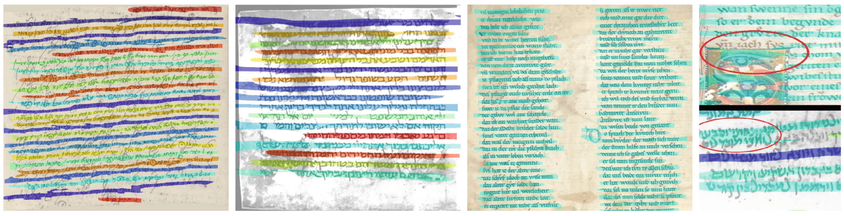


Fig. 5. (a)-(c) Selected result samples of the algorithm: (a) Pinkasim ; (b) Genizah ; (c) Parzival. (d)(upper) The drawing causes the line above it to be missed; (d)(lower) two partial lines accidentally merged together.

are in close vicinity with a text line the result of the algorithm may produce incorrect results, as shown in Fig. 5(d).

6 Conclusions and Future Directions

In this paper, we presented a text line segmentation method for handwritten historical documents. Our approach applies smearing at different scales using a Gaussian scale-space, while utilizing the average height of the characters, followed by a dedicated binarization technique that is based on component-tree and utilize the structure of text lines. In future research we plan to upgrade the proposed method in two directions: (1) refine the use of the evolution maps (EM) to obtain a more reliable range of character heights in the document, and (2) find a more robust procedure for estimating whether a segment refers to a text line or not.

Acknowledgments. This research was supported in part by the DFG-Trilateral grant no. FI 1494/3-2, the Ministry of Science and Technology of Israel, the Council of Higher Education of Israel, the Lynn and William Frankel Center for Computer Sciences and by the Paul Ivanier Center for Robotics and Production Management at Ben-Gurion University, Israel.

References

1. Baechler, M., Liwicki, M., Ingold, R.: Text line extraction using dmlp classifiers for historical manuscripts. In: ICDAR, pp. 1029–1033 (2013)
2. Bar-Yosef, I., Hagbi, N., Kedem, K., Dinstein, I.: Text Line Segmentation for Degraded Handwritten Historical Documents. In: ICDAR, pp. 1161–1165 (2009)
3. Biller, O., Kedem, K., Dinstein, I., El-Sana, J.: Evolution maps for connected components in text documents. In: ICFHR, pp. 403–408 (2012)
4. Bukhari, S.S., Shafait, F., Breuel, T.M.: Script-independent handwritten textlines segmentation using active contours. In: ICDAR, pp. 446–450 (2009)
5. Cohen, R., Asi, A., Kedem, K., El-Sana, J., Dinstein, I.: Robust text and drawing segmentation algorithm for historical documents. In: HIP, pp. 110–117 (2013)
6. Delong, A., Osokin, A., Isack, H.N., Boykov, Y.: Fast approximate energy minimization with label costs. IJCV **96**(1), 1–27 (2012)
7. Diem, M., Kleber, F., Sablatnig, R.: Text line detection for heterogeneous documents. In: ICDAR, pp. 743–747 (2013)
8. Gatos, B., Stamatopoulos, N., Louloudis, G.: ICDAR2009 handwriting segmentation contest. IJDAR **14**(1), 25–33 (2011)
9. Gatos, B., Stamatopoulos, N., Louloudis, G.: ICFHR 2010 handwriting segmentation contest. In: ICFHR, pp. 737–742 (2010)
10. Li, Y., Zheng, Y., Doermann, D., Jaeger, S.: Script-independent text line segmentation in freestyle handwritten documents. IEEE TPAMI **30**(8), 1313–1329 (2008)
11. Lindeberg, T.: Feature detection with automatic scale selection. IJCV **30**(2), 79–116 (1998)
12. Naegel, B., Wendling, L.: A document binarization method based on connected operators. Pattern Recognition Letters **31**(11), 1251–1259 (2010)
13. Rabaev, I., Biller, O., El-Sana, J., Kedem, K., Dinstein, I.: Text line detection in corrupted and damaged historical manuscripts. In: ICDAR, pp. 812–816 (2013)
14. Saabni, R., Asi, A., El-Sana, J.: Text line extraction for historical document images. Pattern Recognition Letters **35**, 23–33 (2014)
15. Shi, Z., Setlur, S., Govindaraju, V.: A steerable directional local profile technique for extraction of handwritten arabic text lines. In: ICDAR, pp. 176–180 (2009)
16. Stamatopoulos, N., Gatos, B., Louloudis, G., Pal, U., Alaei, A.: ICDAR 2013 handwriting segmentation contest. In: ICDAR, pp. 1402–1406 (2013)