Using Scale-Space Anisotropic Smoothing for Text Line Extraction in Historical Documents

Majeed Kassis, Berat Kurar, Rafi Cohen, Jihad El-Sana and Klara Kedem

Department of Computer Science,
Ben-Gurion University, Beer-Sheva, Israel

Abstract

This paper presents a novel approach for text line extraction which is based on Gaussian scale space, a dedicated binarization, and an energy minimization framework. 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 for line extraction. The final stage of the algorithm is based on an energy minimization framework for removing spurious text lines and assigning connected components to lines. We developed two algorithms along these ideas, one for documents with mild curled, generally horizontal lines and the other for multi-skew and curled text lines. We tested our approach on various datasets written in different languages at varying range of image quality and received high detection rates, which outperform state-of-the-art algorithms.

Keywords: Line extraction; Multilingual Handwriting; Energy Minimization; Minimum Spanning Tree;

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

*Corresponding author: majeek@cs.bgu.ac.il

volume 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 [1, 2, 3] 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 [4, 5], and 3rd in ICDAR 2013 contest [6]). However, the performance of these methods depends on choosing the correct scale of the Gaussian filter. Most authors do not provide a scheme to choose the correct scale [1, 2] or determine the scale based on an ad-hoc heuristics [3].

Seam-based line extraction algorithms compute an energy map, which is used to guide the progress of the seam that determines the text lines or their boundaries. These algorithms often fail to detect short text lines and determining the boundary seams of the text-lines is based on an ad-hoc heuristics [7, 8].

In this paper we present a novel approach to detect text lines, which is based on a robust theoretical background; i.e., scale space theory [9]. Our approach applies multi-scale anisotropic second derivative of Gaussian filter bank to enhance the text lines and then applies a dedicated binarization, which is tailored for line extraction in documents. Finally, energy minimization approach is adopted to remove spurious text lines and assign connected components to text lines.

We developed and implemented two algorithms along these ideas, one for general document images that may include slightly curled and skewed horizontal lines, and the other for complicated layouts with multi-skew and highly curled text lines. The first algorithm handles the common case efficiently and the second algorithm tackles the rare complicated case at the cost of a higher complexity. We tested our algorithms on different datasets of various complexities and obtained excellent results. As seen our algorithm outperforms state of art algorithms.

The absence of publicly available benchmark for evaluating multi-skew and curled text line extraction algorithms drove us to develop a dataset, which con-

sists of various historical manuscripts and is publicly available [1]. We manually generated annotations that determine the text lines and their boundaries for these document images.

In the rest of this paper we give a short survey on related works and background literature. We then describe our approach in detail followed by experimental evaluation. Finally we conclude and draw directions for future work. Our MATLAB code is publicly available.[2].

2. Related Work

Extracting handwritten text lines from document images is a basic procedure for various document processing applications and it has received enormous attention over the last several decades. Text line extraction methods can be divided roughly into three classes: top-down, bottom-up, and hybrid. Top-down approaches partition the document image into regions, often recursively, based on various global aspects of an input image. Bottom-up approaches group basic elements, such as pixels or connected components, to form text line patterns. Hybrid schemes combine techniques from top-down and bottom-up classes to yield better results.

2.1. Top-down approaches

He and Downton [16] presented the X-Y cut algorithm, which relies on projections along the X and the Y axes, resulting in a hierarchical tree structure. The Hough transform methodology was exploited for text line segmentation [17, 18, 19]. Likforman-Sulem *et al.* [18] extract the best text line hypothesis in the Hough domain and check the validity of the hypothesis in the image domain. Shapiro *et al.* [17] applied a Hough transform to determine the predefined direction for the projection profile. Image smearing was among the earliest approaches used to determine text lines; Wong *et al.* [20] applied image

[1] `http://www.cs.bgu.ac.il/~rafico/GT.zip`

[2] `http://www.cs.bgu.ac.il/~rafico/LineExtraction.zip`

smearing to binarized printed document images. Under the smearing methods, the Fuzzy Run Length Smoothing Algorithm (RLSA) [21] was introduced. The fuzzy RLSA measure is calculated for every pixel as the maximal extent of the background along the horizontal direction. A new gray-scale image is created based on the RLSA measure and the image is binarized. The text line patterns are extracted from the binarized version. Adaptive local connectivity map technique applies steerable direction filter to reveal text line patterns [22]. Nicolaou and Gatos used local minima tracers, to follow the white-most and black-most paths from one side of a line to the other in order to shred the image into text line areas. Alaei *et al.* [23] used a painting technique to smear the foreground portion of the document image and enhance the separability between the foreground and background for unconstrained handwritten text-lines. Ouwayed and Belaïd [24] used image meshing and the WignerVille distribution to extract multi-oriented lines. A seam-carving-based approach has been developed recently [7, 8]. Saabni *et al.* [7] 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.

2.2. Bottom-up approaches

In contrast to machine printed document images, simple rules such as nearest neighbor do not work well for handwritten documents; e.g., the nearest neighbor often belongs to the next or previous text line. The approaches in this category require the isolation of basic elements, such as strokes and connected components, which is a complicated procedure when components touch across consecutive text lines. OGorman [28] groups components based on the geometric relationship among k-nearest neighbors. Kise *et al.* [31] combine heuristic rules and the Voronoi diagrams to merge connected components into text lines. Nicolas *et al.* [29] adopted the artificial intelligence concept production systems to search for an optimal alignment of connected components into text lines. The minimum spanning tree (MST) clustering technique was used to group components to text lines [32, 27]. Gestalt laws, such as, proximity, similarity, and

direction continuity were also used to iteratively construct lines by grouping neighboring connected components [25]. Garz *et al.* [30] extract Difference of Gaussian (DoG) feature points and cluster words in high-density regions to extract text lines by concatenating neighboring word clusters. Rabaev *et al.* [33] used a sweep-line to aggregate connected components, that correspond to characters, into text lines.

2.3. Hybrid approaches

Among hybrid approaches one can find the work presented recently by Kumar *et al.* [34] where they first suggest a coarse text line estimation, followed by an error detection and correction step to refine text line estimation and assigning misclassified diacritic symbols. Some other methods use active contours techniques for line extraction [1, 35, 3]. Bukhari *et al.* [3] apply a filter bank to smooth the input text image. The central line of text line components is then computed using ridges in the smoothed image. As a final step, active contours (snakes) evolve over the ridges to segment text lines. A dynamic programming based approach, was presented by Liwicki *et al.* [36] for on-line text line segmentation, and adapted by Fischer *et al.* [37] for historical documents. They used histogram analysis, starting points as well as estimated linear line skews, to find a separating path between consecutive lines. This path is required to be smooth, avoid line crossing, and evade strokes below it.

It is important to emphasize that the presence of dots, strokes and diacritics in some scripts such as Greek, Persian and Arabic, complicates text-line extraction. They may cause the Hough transform-based approaches [18, 19] to fail in determining text lines mainly because these approaches consider a whole word and a small stroke as equally important in the Hough domain. Grouping approaches [28, 27] may connect dots with some strokes generating new connected components that may be segmented later as a text line. Smearing approaches [1, 2] either lack a mechanism for determining the appropriate scale of the filter for degraded gray-scale historical documents or choose the scale by binarizing the document and inspecting its height histogram [3], which is suscep-

tible to noise in degraded documents, see Figure 1(a)-(b). Moreover, level-set based active contours methods [1, 35] are computationally complex.

Scale-space ideas were used by Manmatha *et al.* [11] for segmenting words from historical handwritten documents. The authors convolved each text line with a Laplacian of Gaussian (LoG) using an increasing scale and choosing the appropriate scale as the one that maximizes the extent of the generated blob. The authors stated that choosing a larger scale for their algorithm can be used to extract text lines, but they did not develop an algorithm using this approach.

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

3. Notations and Definitions

Our approach relies on scale space scheme and our first algorithm utilizes component tree to extract text lines. To simplify the presentation of the 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 $L : \mathbf{R^2} \times \mathbf{R}_+ \rightarrow \mathbf{R}$ can be defined by convolution with anisotropic Gaussian kernels of varying lengths $\sqrt{t_x}$ and $\sqrt{t_y}$ (the width and the height of the Gaussian, respectively) in the coordinate directions, where $L(x, y; t_x, t_y) = g(x, y; t_x, t_y) * I(x, y)$, and $g : \mathbf{R^2} \times \mathbf{R^2_+} \rightarrow \mathbf{R}$ is an anisotropic Gaussian defined in Eq. (1).

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

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 [9] showed that the amplitude of spatial derivatives, $\partial_{x^\alpha}\partial_{y^\beta} L(x, y; t_x, t_y)$, in general *decreases with scale*; i.e., if an

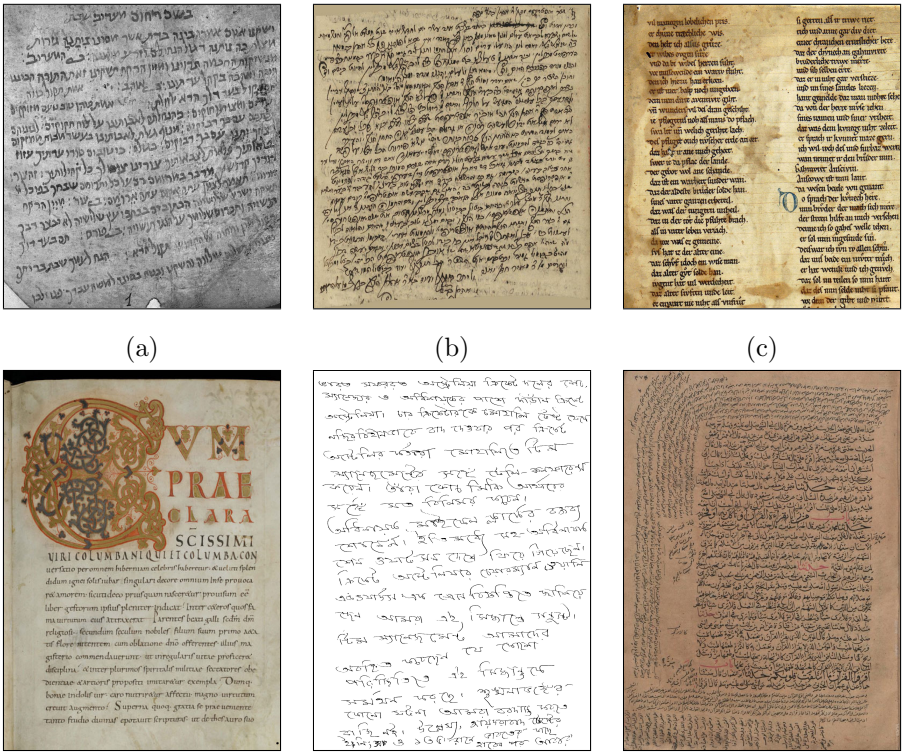


Figure 1: Samples of the documents on which we perform our tests; (a) Genizah handwritten manuscript; (b) Pinkasim handwritten cursive manuscript; (c) German manuscript from Parzival dataset; (d) Latin manuscript from Saint Gall dataset; (e) a manuscript from ICDAR 2013 contest; (f) curled manuscript from the Islamic handwriting collection of Leipzig University.

image is subject to scale-space smoothing, then the numerical values of spatial derivatives computed from the smoothed data can be expected to decrease. He suggested to multiply the derivative with the square root of the scale, i.e., $\partial_\xi = \sqrt{t_x}\partial_x$ to obtain a *scale invariant* function, named the γ-normalized function of the derivatives [9].

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* [38]. 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 the tree construction is simply the maximal value in the map. An illustration can be seen in Figure 2.

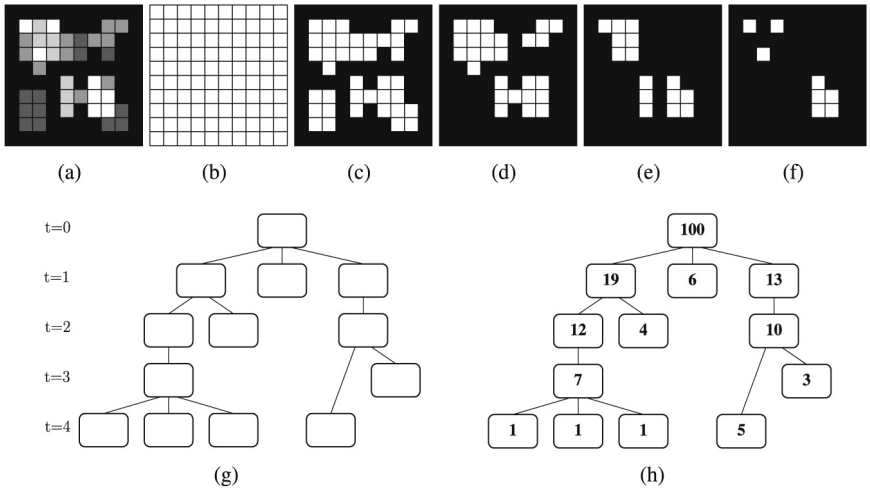


Figure 2: (a) A gray-level image F and its successive threshold sets $B_t(F)$ for t from 0 in (b) to 4 in (f), where $d = 1$; (g) the component-tree of F. (h) The same tree, enriched by an attribute (the size of the connected component of each node), courtesy of [38].

4. Our Approach

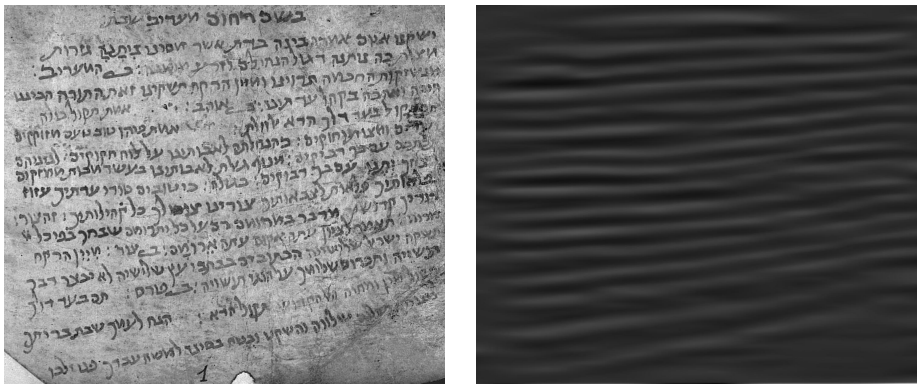
We describe our text line segmentation for handwritten documents. It is based on Gaussian scale space and component-tree traversal. The method starts by enhancing lines in the image, using multi-scale anisotropic second derivative

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 and broken line segments are connected.

4.1. Line enhancement for slight skew

The lines are assumed to be generally horizontal with optional slight skew. The pixels in such an image can be regarded as two dimensional random variables, which are generated by an unknown Probability Distribution Function (PDF). In general, the pixels of the text lines have smaller intensity values than those in the rest of the page. Valleys in the probability map represent text lines and peaks are boundaries between neighboring text lines. Therefore, a convolution of a 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 [39]. Making it an appropriate filter for enhancing slightly skewed horizontal lines in a document.

The appropriate scale for this filter corresponds to the text line height, which varies along the text line itself due to ascenders and descenders, and across text lines. To overcome this limitation, we apply a multi-scale filtering procedure to detect the optimal scale for each point using the scale-space framework [9]. We construct a scale space representation of the images by convolving the image with the γ-normalized function of g_{xx} from Eq. (1) elongated along the x axis, and choose for each pixel the strongest response along the scale-space. We convolve the images with scales that are within the height range of the characters in the document. To estimate the range of character height in gray-scale images, we apply evolution map (EM) tool introduced in our lab by Biller *et al.* [40], which provides details about the height range of the characters in the document, without binarization for grayscale images. 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. Figure 3 illustrates the result of line enhancement on a document from the Genizah collection.



(a) (b)

Figure 3: Applying the line enhancement operator

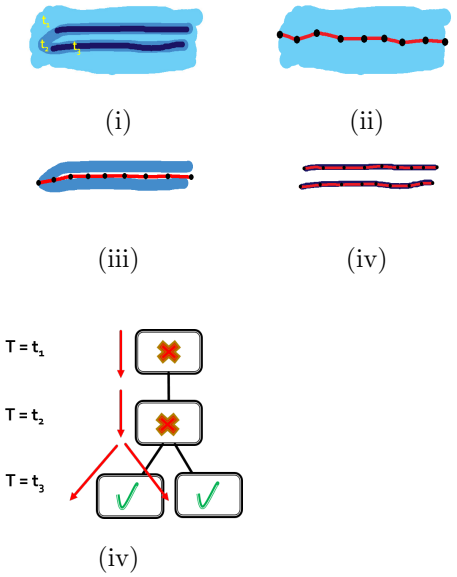
4.2. Text line extraction using component-tree

Applying the Gaussian scale space on the original image (Section 4.1) produces a gray scale image. To extract the text lines we need to binarize the resulting image, $\mathcal{R}$. Off-the-shelf binarization algorithms do not take into account the properties of the resulting image, require tuning parameters, and often introduce noise and other 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 a node 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 get $F(C_i)$ by computing the skeleton of the component and measuring its linearity, but skeleton structure is not robust and is sensitive to noise. Instead, we fit a simple piecewise linear approximation to the connected component and measure the quality, F, of the fit. The mathematical formulation of the piecewise linear approximation is given in Appendix A. To extract the text lines we traverse the component-tree top-down and at each node, C_i, we measure its quality, $F(C_i)$, and based on that we determine

1: $Ouput = \phi$.
2: Enqueue the root 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.
9: **end if**
10: **end while**
11: return $Output$.

(a)



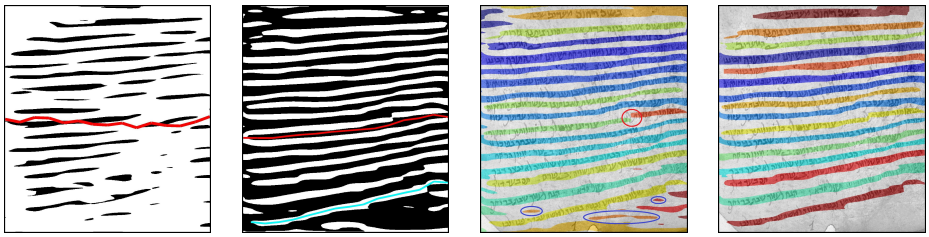
(b)

Figure 4: (a) Pseudo code of the traversal Algorithm; (b) illustration of the algorithm traversal on a synthetic blob; (i) the blob is composed of three gray levels with ascending values, t1, t2 and t3; (ii-iv) the result of thresholding the blob with $T \geq t_1, t_2, t_3$ respectively, and the approximating spline; (iv) BFS traversal of the component tree.

whether it represents a text line or not. If C_i represents 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. 4(a) presents the pseudo-code of the traversal procedure.

4.3. Energy-based formulation of text-line extraction problem

The algorithm in Figure 4 generates a set of potential text lines (Figure 5(c)). Our goal is to find a consistent segmentation of connected components (CCs) into lines while using an optimal subset of the potential text lines. To accomplish this we use the energy minimization framework which is a powerful and popular framework for solving labeling problems in computer vision [41]. Here, we present an energy function based on a binarized version of the document, whose minimization yields an assignment of connected-components (characters)



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

Figure 5: The various stages of the component tree traversal, where splines are 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.

to text lines and removal of spurious lines that do not correspond to text lines. Our function consists of three terms: data, smoothness, and label costs. The data cost measures how likely is a connected component part of a particular line, whereas the smoothness term is responsible for spatial coherence, i.e., CCs with a smaller distance have a much higher probability to belong to the same line than those distant ones. The label cost penalizes overly-complex models and favors the explanation of the document with fewest and cheapest lines.

More formally, let $\mathcal{L}$ be the set of lines (labels and lines are interchangeable throughout this section) that were extracted in Section 4.2 (Fig. 5(c)) and let $\mathcal{C}$ be the set of connected component (binarized characters) 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.

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

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 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 the Euclidean distance [42]). 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 [39]. The term $\delta(\ell_c \neq \ell_{c'})$ is Kronecker's delta [43].

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. The label cost h_ℓ is defined as $\exp(\gamma_1 \cdot r_\ell)$, where r_ℓ is the normalized number of foreground pixels in the binarized document overlapping with ℓ and γ_1 is a constant we set experimentally. This label cost penalizes the use of lines that have a small overlap with background pixels compared to other lines.

4.4. Joining broken line segments

Our algorithm usually extracts the correct text line efficiently. However, in some cases it splits a text lines into disconnected segments, as shown in Figure 5(c). We overcome these limitations by detecting and connecting segments that belong to the same text line. 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 components 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. In case a connected component c, overlaps two neighboring text lines, we relabel c by assigning each pixel in c to the closest line.

5. The Generalized Algorithm for Curled Lines

In this section we adapt our algorithm for handling highly curled and multi-oriented text lines. A typical example of such text lines are the side-notes written in page margins as remarks on the main-text. While the main-text lines are mostly horizontally oriented side-note lines can be curled and may have different orientations and locations on the page margins, as shown in Figure 1(f). Accurate segmentation of these lines is a challenging task that has great importance for many applications. For our study we separate the main-text from the side-notes text, using graph cuts [44] and focus on the side-notes. To the best of our knowledge, this is the first work designated to segment text lines with severe curvature in the context of historical documents.

Enhancement of the text lines is done by convolving the image with the second derivative of the Gaussian as explained in Section 4.1, but with varying orientations, we use all the orientations between 0° and 180°, with a fixed step size. Using such a large range of orientations may lead to false ligatures introduced between lines (see Figure 6(a)), a fact which poses a real difficulty to our binarization algorithm. Therefore, a different strategy is applied. We use a standard adaptive binarization (Niblack [45]), followed by decomposition of the invalid text lines into line parts and the removal of the false ligatures. The classification of text lines into valid and invalid ones, is based on how well a line can be approximated by a piecewise linear approximation. The decomposition is based on the morphological skeleton, where the junction points of the skeleton separate between different parts of the decomposed line (see Figure 6(b)). After the decomposition phase, an energy minimization technique is used to remove false ligatures, which is followed by merging of broken line segments. In the final stage, the pixels in the connected components are assigned to lines and spurious lines are removed by energy minimization. In the rest of this section we discuss in detail the required changes to the previously presented algorithm.

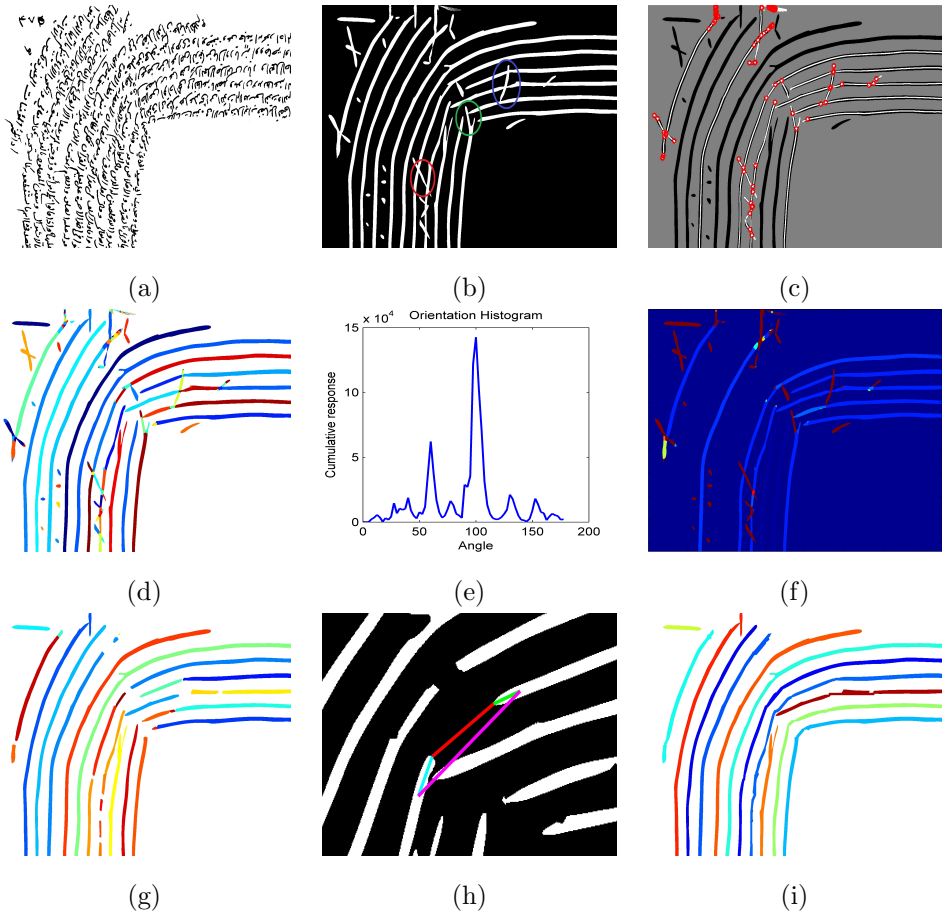


Figure 6: An overview of line extraction on the patch given in (a). (b) After applying the multi-oriented smearing operator and Niblack's binarization. Some false ligatures are encircled. (c) Invalid lines are decomposed into adjacent smaller line parts based on the junction points of the skeleton; the skeleton is overlaid on the invalid lines (in white) and the junction points are encircled in red. (d) Different line part are encoded in different colors. (e) The orientation histogram of the line part encircled in red from (b), the peak is located at 100°. (f) The label cost for the line parts. (g) The result after energy-minimization, different line parts are encoded using different colors. (h) Computation of the edge weight between two end-points, explained in the text. (i) The final result after merging line parts.

5.1. Energy-based formulation of text-line extraction

Our text-line extraction consists of four parts: a) removing false ligatures between lines; b) merging broken line segments; c) assigning connected compo-

nents (characters) to lines, and d) removing spurious lines.

We compute for each line part resulting from an invalid line, a label cost that describes how much its orientation deviates from the dominant orientation concentrated in a small radius around it. Line parts with high cost will be removed using the energy minimization framework in similar way to Section 4.3. The computation of the dominant orientation uses an orientation histogram in the following way. The bins in the histogram are the different orientations used to filter the image (Section 4.1). If a pixel in the area responded most strongly to a filter with orientation θ_1, the absolute value of the response is added to the bin belonging to θ_1. The peak of this orientation histogram, θ_{his}, is chosen to represent the dominant orientation of that area. Figure 6(d) depicts the orientation histogram of the line part encircled in red in Figure 6(b). We also compute for each line part its orientation based on PCA, θ_{PCA}, and assign a high label cost for the line part if the two computed orientations deviate much from each other by using Eq. (3), where γ_2 is a constant. The label costs are shown in Figure 6(e) and the result after the energy minimization is depicted in Figure 6(f), the false ligatures were removed but also some text lines were disconnected.

$$h_\ell = \exp(\gamma_2 \cdot (1 - |\cos(\theta_{PCA} - \theta_{his})|)). \tag{3}$$

5.1.1. Merging broken line segments using MST

For deciding the broken lines to be merged we use a minimum spanning tree (MST) in an undirected weighted graph. Our graph $G = (V, E)$ is defined as follows: the set of vertices, V, is composed of all the end-points of the currently detected lines, in addition we have a root vertex that is connected to all the end-points. The set of edges is $E = E_1 \cup E_2 \cup E_3$, where E_1 is the set of edges that belong to the same line, their weight is set to 0. The set E_2 is the set of edges between the root and all the vertices, their weight, $w(u, v)$, is the normalized number of foreground pixels overlapping with the adjacent line, where the goal is to encourage small lines to connect to larger text lines. The set E_3 consists of edges between end-points that belong to different lines. The weight of $(u, v) \in$

E_3 is a local linearity measure, which is defined below. We do not consider edges such that a) overlap different lines, and b) edges that generated lines that are not well approximated by a piecewise linear approximation. Minimizing a bounded degree spanning tree is known to be NP-hard [46], so instead we solve the following approximation to the problem. We generate a graph where each vertex u, except for the root has degree two. This is done by connecting u to a vertex v, such that the curve generated by (v, u) is the most successful locally straight line curve, which is defined below.

We measure how well the connection between u and v generates a curve that is locally a straight line, for each vertex (endpoint), u or v we choose a nearby point on the line to it belongs. Let us denote these points by s and t, respectively. The local linearity measure is defined in Eq. (4), where γ_3 is a constant and $||u - v||$ is the Euclidean distance between u and v. This computation is illustrated in Figure 6)(h), where the lines between s and u, u and v, v and t, and s and t are colored in cyan, red, green and purple respectively. The result of this procedure is illustrated in Figure 6(i), which shows, the result of merging disconnected lines according to the MST computation.

$$w(u, v) = \exp\left(\gamma_3 \cdot \left(\frac{||s - u|| + ||u - v|| + ||v - t||}{||s - t||} - 1\right)\right). \tag{4}$$

6. Experimental results and Discussions

We evaluated our first text line detection algorithm 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 [6], ICDAR 2009 [4], Hebrew [33], Saint Gall [47], Parzival [48] and BGU [7] 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. The BGU dataset contains 315 pages written in Arabic, English and Chinese both in gray-scale and in a binary format. For Parzival we used the ground-truth generated by [33].

The performance evaluation is based on a MatchScore [6] that computes the maximum overlap of a text region with the ground truth region. If this score is above a given threshold T_α then the text line is considered correct (one-to-one match, o2o [6]). Based on this MatchScore, the Detection Rate (DR), the Recognition Accuracy (RA), and the Performance Metric (FM) are defined using Eq. (5), 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 to 95% for datasets of binary images, and 90% for datasets of gray-scale images. 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 [47, 48]. The results of the presented algorithm are reported in Table 1, where it is evident that our algorithm outperformed state-of-the-art algorithms. We also mention for each dataset whether it consists of binary pages (B) or gray-scale pages (G).

$$DR = \frac{o2o}{N}, RA = \frac{o2o}{M}, FM = \frac{2 \times DR \times RA}{DR + RA} \tag{5}$$

Throughout the experiments we use 20 knots (K=20) to measure the linearity of the components, the scale space is defined based on $d = 1$ and $\sqrt{\frac{t_x}{t_y}} = 3$, and the fixed step size used in generating the different orientations is $\Delta\theta = 2.5°$. Using a standard laptop equipped with an Intel Core i7 processor our unoptimized Matlab implementation runs at 41.2 seconds on an average ICDAR 2013 document.

Although the algorithm achieves high detection rates, it has some limitations. For example, if salient objects in the image, such as holes and drawings, are in proximity to a text line the result of the algorithm may produce incorrect results, as shown in Figure 7(d).

The absence of publicly available benchmark for evaluating multi-skew and curled text line extraction algorithms (e.g., Figure 1(f)) drove us to develop our own dataset[3], which consists of 25 historical manuscripts with 261 text lines written in Arabic from the Islamic Heritage Project (IHP), Harvard. The images were selected to have challenging multi-skew and highly curled text lines. The groundtruth annotation for the dataset has been manually generated by labeling each connected component, where overlapping components were relabeled at pixel level. The documents in the dataset consist of a mixture of main-text and side-notes, to test our algorithm we focused on the side-notes only, as they are curled with different size and skew. See Figure 8(a)-(c) for an example of these documents.

The side-notes in these documents contain a variety of text lines of different lengths, some text lines are of length of one or two words (See Figure 8), whereas some are much longer. In the traditional evaluation protocol, both text lines contributed equally to the evaluation, which made it an unfair evaluation, since the aforementioned short lines are (a) harder to segment correctly, and (b) contain very few words. So, in our evaluation we normalize the contribution of each line with respect to its length. Let $\mathcal{L}$ be the set of text lines returned by *our algorithm*, and $\mathcal{L}_{GT}$ the set of lines in the ground truth. For $\ell \in \mathcal{L}$ and $\ell' \in \mathcal{L}_{GT}$, we define r_ℓ and r'_ℓ, as the normalized number of foreground pixels overlapping with ℓ and ℓ' respectively. Then, the Normalized Detection Rate (NDR), and the Normalized Recognition Accuracy (NRA) are defined in Eq. (6). The results of the presented algorithm are $NDR = 77.99\%$, $NRA = 77.95\%$ and $FM = 77.97\%$.

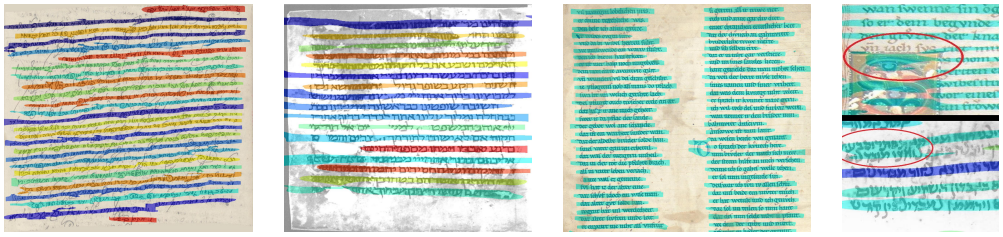
$$NDR = \frac{\sum_{\ell \in o2o} r_\ell}{\sum_{\ell \in \mathcal{L}} r_\ell}, \quad NRA = \frac{\sum_{\ell \in o2o} r'_\ell}{\sum_{\ell \in \mathcal{L}_{GT}} r'_\ell} \tag{6}$$

To further test our method, we manually annotated 37 document pages containing various curved side note lines to illustrate further the strength of the algorithm. A sample image, and its annotated

[3] `http://www.cs.bgu.ac.il/~rafico/GT.zip`

	Our Method					state-of-the-art	
	M	o2o	DR	RA	FM	FM	
ICDAR 2013 [6](B)	2651	2621	98.94%	98.86%	**98.90%**	98.66%	
ICDAR 2009 [4](B)	4033	4021	99.67%	99.70%	**99.69%**	99.53%	
Hebrew [33](G)	1257	1154	89.04%	91.88%	**90.44%**	86.10%	
BGU [7](B+G)	5403	5303	98.31%	98.15%	**98.23%**	98.0%	
	PHR			FM		PHR	FM
Saint Gall [47](G)	**99.08%**			**99.22%**		98.94%	99.03%
Parzival [48](G)	**98.31%**			**97.88%**		96.30%	96.40%

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).



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

Figure 7: (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.

lines can be seen in Figure 9. The algorithm managed to achieve a strong result of 98.97% **accuracy on the dataset with** $k = 20$ **on the line level. We also applied a comparison of pixel-level comparison using XOR, and resulted in comparison error rate of** 12.87%**, or** 87.13% **of pixel to pixel accuracy. The dataset is publicly available on our VML website [49] for free search use.**

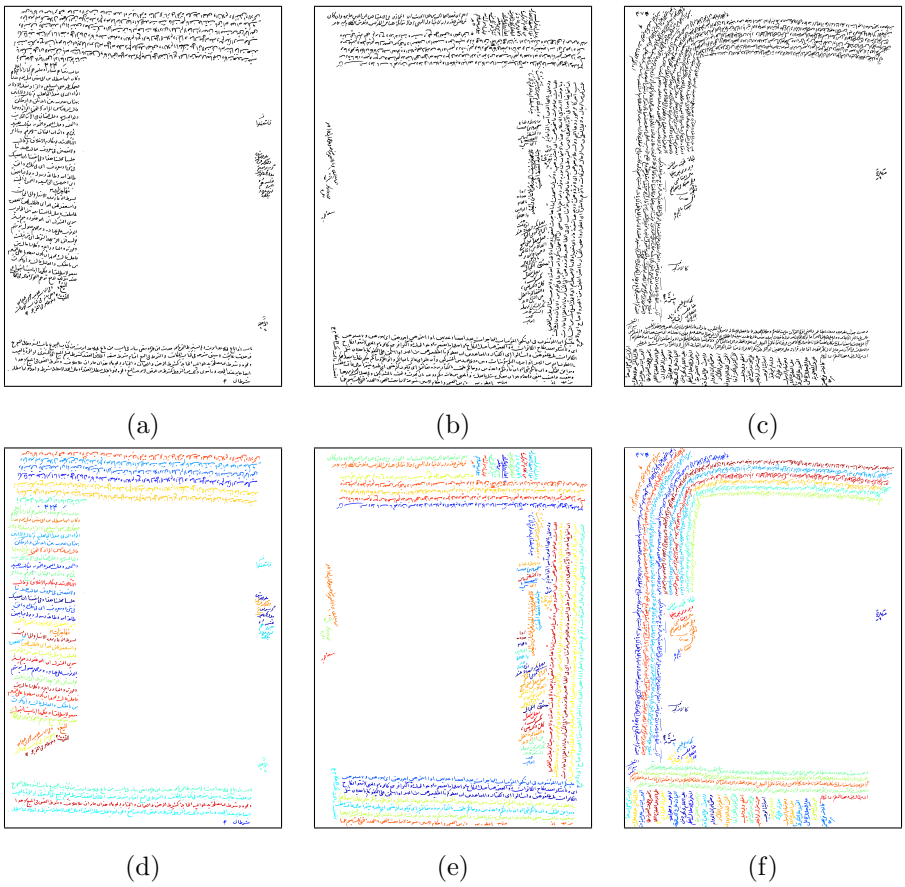


Figure 8: (a)-(c) Samples of the documents on which we perform our tests; (d)-(f) the corresponding results of our algorithm.

7. 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 takes into account 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

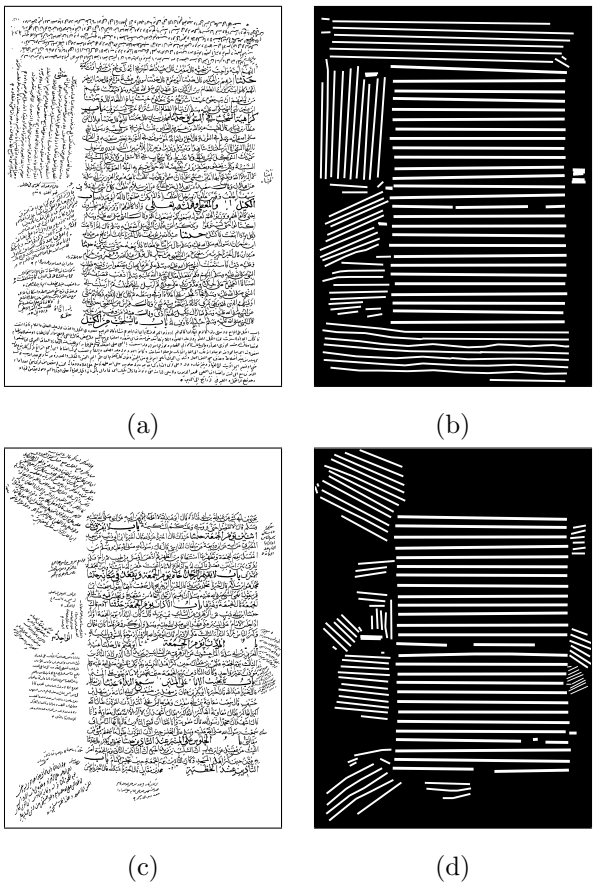


Figure 9: (a),(c) two sample images. (b),(d) the annotated line mask of the two images.

segment belongs to a text line or not. (3) We also plan to divide the multi-skew documents into separate regions, and utilize this division to improve our results. Our MATLAB code is publicly available.[4]

Appendix A. Piecewise linear approximation of connected-components

We denote the spatial location of pixels belonging to a connected-component C as (x_i, y_i), $i = 1, \ldots, m$. We assume that x_i are sorted, i.e., $x_1 \leq x_2 \leq \cdots \leq x_m$. Let $a_0 < a_1 < a_2 < \cdots < a_K$ be a set of fixed equal-distance points

[4] `http://www.cs.bgu.ac.il/~rafico/LineExtraction.zip`

(knots), with $a_0 = x_1$ and $a_K = x_m$. Our goal is to find a piecewise linear function f, defined over $[a_0, a_K]$, with knot points a_i, that minimizes a least-squares fitting criterion. One method to solve this problem is based on defining $f(x) = \alpha_i x + \beta_i$, for all $x \in (a_{i-1}, a_i]$ and adding conditions on the parameters α_i and β_i to ensure that f is continuous. This is formulated in Eq. (A.1), which can be reformulated into a Quadratic Programing (QP) by representing $f(x_i)$ in matrix form, Eq. (A.2), where the variables are $\alpha \in \mathbf{R}^K$ and $\beta \in \mathbf{R}^K$, the data are $x \in \mathbf{R}^m$, $y \in \mathbf{R}^m$, F_{ij} is defined in Eq. (A.3) and **diag(x)** is a square diagonal matrix with the elements of vector x on the main diagonal. This QP can be solved by standard tools as CVX [50]. Let us denote the interval $(a_{i-1}, a_i]$ as I_i and the number of points from C, with an abscissa in I_i as n_i. The fitness $F(C)$ is defined as the highest average distance of points from C to f, among all intervals, which is formulated in Eq. (A.4).

$$\begin{aligned} &\text{minimize} && \sum_{i=1}^{m} (f(x_i) - y_i)^2 \\ &\text{subject to} && \alpha_i a_i + \beta_i = \alpha_{i+1} a_i + \beta_{i+1},\ i = 1, \ldots, K-1. \end{aligned} \tag{A.1}$$

$$\begin{aligned} &\text{minimize} && ||\mathbf{diag}(\mathbf{x}) F\alpha + F\beta - y||^2 \\ &\text{subject to} && \alpha_i a_i + \beta_i = \alpha_{i+1} a_i + \beta_{i+1},\ i = 1, \ldots, K-1, \end{aligned} \tag{A.2}$$

$$F_{ij} = \begin{cases} 1 & \text{if } a_{j-1} = x_i, j = 1 \\ 1 & \text{if } a_{j-1} < x_i \leq a_j \\ 0 & \text{otherwise} \end{cases} \tag{A.3}$$

$$F(C) = \max_i \frac{1}{n_i} \sum_{x_j \in I_i} (f(x_j) - y_j)^2. \tag{A.4}$$

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] Y. Li, Y. Zheng, D. Doermann, S. Jaeger, Script-Independent Text Line Segmentation in Freestyle Handwritten Documents, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 30 (8) (2008) 1313–1329.

[2] Z. Shi, S. Setlur, V. Govindaraju, A steerable directional local profile technique for extraction of handwritten arabic text lines, in: The International Conference on Document Analysis and Recognition (ICDAR), 2009, pp. 176–180.

[3] S. S. Bukhari, F. Shafait, T. M. Breuel, Coupled snakelets for curled text-line segmentation from warped document images, International Journal on Document Analysis and Recognition (IJDAR) 16 (1) (2013) 33–53.

[4] B. Gatos, N. Stamatopoulos, G. Louloudis, ICDAR2009 handwriting segmentation contest, International Journal on Document Analysis and Recognition (IJDAR) 14 (1) (2011) 25–33.

[5] B. Gatos, N. Stamatopoulos, G. Louloudis, ICFHR 2010 handwriting segmentation contest, in: The International Conference on Frontiers in Handwriting Recognition (ICFHR), 2010, pp. 737–742.

[6] N. Stamatopoulos, B. Gatos, G. Louloudis, U. Pal, A. Alaei, ICDAR 2013 handwriting segmentation contest, in: The International Conference on Document Analysis and Recognition (ICDAR), 2013, pp. 1402–1406.

[7] R. Saabni, A. Asi, J. El-Sana, Text line extraction for historical document images, Pattern Recognition Letters 35 (0) (2014) 23 – 33.

[8] N. Arvanitopoulos Darginis, S. Süsstrunk, Seam carving for text line extraction on color and grayscale historical manuscripts, in: The International Conference on Frontiers in Handwriting Recognition (ICFHR), 2014, pp. 726–731.

[9] T. Lindeberg, Feature detection with automatic scale selection, International Journal of Computer Vision (IJCV) 30 (2) (1998) 79–116.

[10] M. R. Hashemi, O. Fatemi, R. Safavi, Persian cursive script recognition, in: The International Conference on Document Analysis and Recognition (ICDAR), 1995, pp. 869–873.

[11] R. Manmatha, J. Rothfeder, A scale space approach for automatically segmenting words from degraded handwritten documents, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 27 (8) (2005) 1212–1225.

[12] V. Bosch, A. H. Toselli, E. Vidal, Statistical text line analysis in handwritten documents, in: The International Conference on Frontiers in Handwriting Recognition (ICFHR), 2012, pp. 201–206.

[13] M. Arivazhagan, H. Srinivasan, S. N. Srihari, A Statistical Approach to Handwritten Line Segmentation, in: Document Recognition and Retrieval XIV, Proceedings of SPIE, 2007, pp. 1–11.

[14] A. Zahour, L. Likforman-Sulem, W. Boussalaa, B. Taconet, Text line segmentation of historical arabic documents, in: The International Conference on Document Analysis and Recognition (ICDAR), 2007, pp. 138–142.

[15] I. Bar-Yosef, N. Hagbi, K. Kedem, I. Dinstein, Text Line Segmentation for Degraded Handwritten Historical Documents, in: The International Conference on Document Analysis and Recognition (ICDAR), 2009, pp. 1161–1165.

[16] J. He, A. C. Downton, User-assisted archive document image analysis for digital library construction, in: The International Conference on Document Analysis and Recognition (ICDAR), 2003, pp. 498–502.

[17] V. Shapiro, G. Gluhchev, V. Sgurev, Handwritten document image segmentation and analysis, Pattern Recognition Letters 14 (1) (1993) 71–78.

[18] L. Likforman-Sulem, A. Hanimyan, C. Faure, A hough based algorithm for extracting text lines in handwritten documents, in: The International Conference on Document Analysis and Recognition (ICDAR), 1995, pp. 774–777.

[19] G. Louloudis, B. Gatos, I. Pratikakis, C. Halatsis, Text line detection in handwritten documents, Pattern Recognition 41 (12) (2008) 3758–3772.

[20] K. Y. Wong, R. G. Casey, F. M. Wahl, Document analysis system, IBM journal of research and development 26 (6) (1982) 647–656.

[21] Z. Shi, V. Govindaraju, Line separation for complex document images using fuzzy runlength, in: International Workshop on Document Image Analysis for Libraries (DIAL), 2004, pp. 306–312.

[22] A. Nicolaou, B. Gatos, Handwritten text line segmentation by shredding text into its lines, in: The International Conference on Document Analysis and Recognition (ICDAR), 2009, pp. 626–630.

[23] A. Alaei, U. Pal, P. Nagabhushan, A new scheme for unconstrained handwritten text-line segmentation, Pattern Recognition 44 (4) (2011) 917–928.

[24] N. Ouwayed, A. Belaïd, A general approach for multi-oriented text line extraction of handwritten documents, International Journal on Document Analysis and Recognition (IJDAR) 15 (4) (2012) 297–314.

[25] L. Likforman-Sulem, C. Faure, Extracting text lines in handwritten documents by perceptual grouping, Advances in handwriting and drawing: a multidisciplinary approach (1994) 117–135.

[26] Y. Pu, Z. Shi, A natural learning algorithm based on hough transform for text lines extraction in handwritten documents, in: The International Workshop on Frontiers in Handwriting Recognition (IWFHR), 1998, pp. 637–646.

[27] F. Yin, C.-L. Liu, Handwritten chinese text line segmentation by clustering with distance metric learning, Pattern Recognition 42 (12) (2009) 3146–3157.

[28] L. O'Gorman, The document spectrum for page layout analysis, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 15 (11) (1993) 1162–1173.

[29] S. Nicolas, T. Paquet, L. Heutte, et al., Text line segmentation in handwritten document using a production system, in: the International Workshop on Frontiers in Handwriting Recognition (IWFHR), Vol. 4, 2004, pp. 245–250.

[30] A. Garz, A. Fischer, H. Bunke, R. Ingold, A binarization-free clustering approach to segment curved text lines in historical manuscripts, in: The International Conference on Document Analysis and Recognition (ICDAR), 2013, pp. 1290–1294.

[31] K. Kise, A. Sato, M. Iwata, Segmentation of page images using the area voronoi diagram, Computer Vision and Image Understanding 70 (3) (1998) 370–382.

[32] A. Simon, J.-C. Pret, A. P. Johnson, A fast algorithm for bottom-up document layout analysis, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 19 (3) (1997) 273–277.

[33] I. Rabaev, O. Biller, J. El-Sana, K. Kedem, I. Dinstein, Text line detection in corrupted and damaged historical manuscripts, in: The International Conference on Document Analysis and Recognition (ICDAR), 2013, pp. 812–816.

[34] J. Kumar, L. Kang, D. Doermann, W. Abd-Almageed, Segmentation of handwritten textlines in presence of touching components, in: The International Conference on Document Analysis and Recognition (ICDAR), 2011, pp. 109–113.

[35] X. Du, W. Pan, T. D. Bui, Text line segmentation in handwritten documents using mumford–shah model, Pattern Recognition 42 (12) (2009) 3136–3145.

[36] M. Liwicki, E. Indermuhle, H. Bunke, On-line handwritten text line detection using dynamic programming, in: The International Conference on Document Analysis and Recognition (ICDAR), 2007, pp. 447–451.

[37] A. Fischer, E. Indermühle, H. Bunke, G. Viehhauser, M. Stolz, Ground truth creation for handwriting recognition in historical documents, in: Proceedings of the 9th IAPR International Workshop on Document Analysis Systems (DAS), 2010, pp. 3–10.

[38] B. Naegel, L. Wendling, A document binarization method based on connected operators, Pattern Recognition Letters 31 (11) (2010) 1251–1259.

[39] R. Cohen, A. Asi, K. Kedem, J. El-Sana, I. Dinstein, Robust text and drawing segmentation algorithm for historical documents, in: International Workshop on Historical Document Imaging and Processing (HIP), 2013, pp. 110–117.

[40] O. Biller, K. Kedem, I. Dinstein, J. El-Sana, Evolution maps for connected components in text documents, in: The International Conference on Frontiers in Handwriting Recognition (ICFHR), 2012, pp. 403–408.

[41] A. Delong, A. Osokin, H. N. Isack, Y. Boykov, Fast approximate energy minimization with label costs, International Journal of Computer Vision (IJCV) 96 (1) (2012) 1–27.

[42] M. Kubovy, M. van den Berg, The whole is equal to the sum of its parts: A probabilistic model of grouping by proximity and similarity in regular patterns, Psychological review 115 (1) (2008) 131–154.

[43] Y. Boykov, O. Veksler, R. Zabih, Fast approximate energy minimization via graph cuts, Pattern Analysis and Machine Intelligence, IEEE Transactions on 23 (11) (2001) 1222–1239.

[44] A. Asi, R. Cohen, K. Kedem, J. El-Sana, I. Dinstein, A Coarse-to-Fine Approach for Layout Analysis of Ancient Manuscripts, in: The International Conference on Frontiers in Handwriting Recognition (ICFHR), 2014, pp. 140–145.

[45] W. Niblack, An introduction to digital image processing, Strandberg Publishing Company, 1985.

[46] M. Singh, L. C. Lau, Approximating minimum bounded degree spanning trees to within one of optimal, in: Proceedings of the thirty-ninth annual ACM symposium on Theory of computing, ACM, 2007, pp. 661–670.

[47] M. Diem, F. Kleber, R. Sablatnig, Text line detection for heterogeneous documents, in: The International Conference on Document Analysis and Recognition (ICDAR), 2013, pp. 743–747.

[48] M. Baechler, M. Liwicki, R. Ingold, Text line extraction using dmlp classifiers for historical manuscripts, in: The International Conference on Document Analysis and Recognition (ICDAR), 2013, pp. 1029–1033.

[49] Visual media laboratory - arabic historical documents dataset, `http://www.cs.bgu.ac.il/~vml`, [Online; accessed 2017] (2016).

[50] M. Grant, S. Boyd, CVX: Matlab software for disciplined convex programming, version 2.1, `http://cvxr.com/cvx` (Mar. 2014).