Stable Text Line Detection

Jaime S. Cardoso
INESC Porto, Faculdade de Engenharia, Universidade do Porto
jaime.cardoso@inescporto.pt

Abstract

Text line segmentation in freestyle handwritten documents remains an open document analysis problem. Curvilinear text lines and small gaps between neighbouring text lines present a challenge to algorithms developed for machine-printed or hand-printed documents. We investigate a general-purpose, knowledge-free method for the automatic detection of text lines based on a stable path approach. Lines affected by curvature and inclination are robustly detected. The proposed methodology was tested on a modern set of handwritten images made available on the ICDAR 2009 handwriting segmentation competition, with promissing results.

1. Introduction

Text line segmentation is one of the major components of document image analysis. It provides crucial information for skew correction, zone segmentation, and character recognition. Although text line segmentation for machine printed or hand-printed documents is usually seen as a solved problem, freestyle handwritten text lines still present a significant challenge. Handwritten document analysis is difficult mainly due to the irregularity of layout and character shapes originated from the variability of writing styles. Handwritten text lines are often un-uniformly skewed and curved, and the space between lines is not obvious. Linear or piecewise linear approximation is not accurate in general. For unconstrained handwritten documents, text line segmentation and character segmentation-recognition are not solved though enormous efforts have been devoted to them and great advances have been made [3, 5, 7].

1.1. Related Works

There are mainly three basic categories that text line segmentation methods lie in: methods making use of the projection profiles, methods that are based on the Hough transform and, finally, smearing methods. In [4, 5] the authors review the related work on the text line segmentation problem.

In spite of the variety of methods available, they all suffer from some limitations. A problem common is that current techniques they do not properly incorporate global information in the detection process. To our knowledge, none of the proposed methods in the literature tries to define a reasonable process from the intrinsic properties of text lines, namely the fact that they are extensive black objects on the document. Generally, we argue that the most interesting techniques arise when one defines the detection process as the result of optimizing some global function. In the following, we suggest a graph-theoretic framework where text lines are the solutions of a global optimization process.

2. A Stable Path Approach for Text Line Detection

The proposed algorithm is an adaptation of an algorithm for staff line detection in a music score [2]. In the work to be detailed, the image grid is considered as an weighted graph with pixels as nodes and arcs connecting neighbouring pixels. The weight of each arc, $w(p, q)$, is a function of pixels values and pixels relative positions.

2.1. Algorithm outline

To a first approximation, text lines can be considered as the only extensive objects made from black pixels in the text document, nearly connected paths of black pixels from the left side to the right side of the image. Assuming that paths through black pixels are preferred over paths through white pixels, lines can then be found among the shortest paths from the left to the right margin of the image. Text lines are then best modelled as paths between two regions Ω_1 and Ω_2, the left and right margins of the document.

One may assume that text lines do not zigzag back and forth, left and right. Therefore, one may restrict the search among connected paths containing one, and only one, pixel in each column of the image[1]. Formally, let I be an $N_1 \times N_2$

[1] These assumptions, 8-connectivity and one pixel per column, impose a maximum detectable 45 rotation degree.

978-1-4244-5496-9/09/$25.00 ©2009 IEEE

image and define an admissible line to be

$$\mathbf{s} = \{(x, y(x))\}_{x=1}^{N_1}, \text{ s.t. } \forall x \, |y(x) - y(x-1)| \leq 1,$$

where y is a mapping $y : [1, \cdots, N_1] \rightarrow [1, \cdots, N_2]$. That is, a text line is an 8-connected path of pixels in the image from left to right, containing one, and only one, pixel in each column of the image.

Given the weight function $w(p, q)$, the cost of a line can be defined as $C(\mathbf{s}) = \sum_{i=2}^{N_1} w(v_{i-1}, v_i)$. The optimal text line that minimizes this cost can be found using dynamic programming. The first step is to traverse the image from the second column to the last column and compute the cumulative minimum cost C for all possible connected text lines for each entry (i, j):

$$C(i,j) = \min \begin{cases} C(i-1, j-1) & + & w(p_{i-1,j-1}; p_{i,j}) \\ C(i-1, j) & + & w(p_{i-1,j}; p_{i,j}) \\ C(i-1, j+1) & + & w(p_{i-1,j+1}; p_{i,j}) \end{cases},$$

where $w(p_{i,j}; p_{l,m})$ represents the weight of the edge incident with pixels at positions (i, j) and (l, m). At the end of this process,

$$\min_{j \in \{1, \cdots, N_2\}} C(N_1, j)$$

indicates the end of the minimal connected line. Hence, in the second step, one backtrack from this minimum entry on C to find the path of the optimal line.

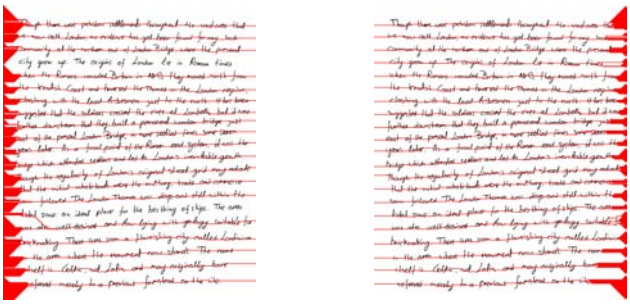
Assume one wants to find all text lines present in a score. This could be approached by successively finding and erasing the shortest path from the left to the right margin of the score. The erase operation is required to ensure that a line is not detected multiple times. A different approach is to try to detect multiple text lines in a single iteration, instead of sequentially computing them one at a time. We summarize now the concept of a stable path in a graph [2], which will allow us to do precisely that.

Definition. A path $\mathcal{P}_{s,t}$ is a stable path between regions Ω_1 and Ω_2 if $\mathcal{P}_{s,t}$ is the shortest path between $s \in \Omega_1$ and the whole region Ω_2, and $\mathcal{P}_{s,t}$ is the shortest path between $t \in \Omega_2$ and the whole region Ω_1.

Note that the concept of stable path is valid for any graph and any two sub-graphs in general.

In Fig. 1(a) the shortest paths between *each point* on the left margin and the *whole* right margin are traced for the score in Fig. 2(a). As seen, the paths got attracted by the text lines. Likewise, Fig. 1(b) shows the shortest paths between *each point* on the right margin and the *whole* left margin. The set of stable paths between both margins result as the set of paths present in both figures.

Although the computation of the stable paths may be expensive in general graphs, the computation in the graph derived from an image under the setting adopted in this section has only roughly twice the complexity of the shortest path



(a) shortest paths from each pixel in the left column and the whole right column, superimposed on the original image.

(b) shortest paths from each pixel in the right column and the whole left column, superimposed on the original image.

Figure 1. Illustration of stable paths for Fig. 2(a).

computation. Noticing that the procedure delineated for the shortest path actually gives the shortest path between the whole left margin Ω_1 and each point on the right margin Ω_2, the first step on the computation of the stable path corresponds verbatim to the computation of the shortest path presented on Section 2.1. In a second step one repeats the same procedure, traversing now the graph from the right column to the left. At the end of this process, if the two endpoints of a direct and reverse path coincide, we are in the presence of a stable path.

Next, the complete proposed algorithm for text line detection is detailed.

2.2. Proposed Algorithm

The proposed algorithm can be implemented as a sequence of a few high-level operations, as presented in Listing 1.

```
PreProcessing:
    image width reduction
    compute weights of the graph
Main Cycle:
   compute stable paths
   validate paths with blackness
   erase valid paths from image
   add valid paths to list of textlines
   end of cycle if no valid path was found

PostProcessing:
   uncross textlines
   smooth textlines
   find separation-path between consecutive text lines
```

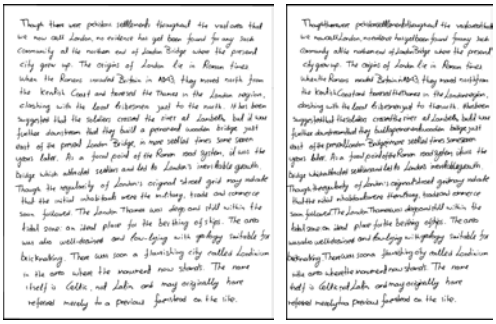
Listing 1: Main operations of the proposed method.

2.2.1 Preprocessing

The white margins on the document and the gaps between words behave like 'noise' to the algorithm, and do not con-

stitute any useful information. Therefore, the proposed algorithm starts by reducing the width of the image with the elimination of vertical paths through white pixels only [1,6]. The procedure to resize the image is totally equivalent to the process used to find the text lines. One searches for the shortest path between the top and bottom margins, this time penalizing paths through black pixels. The resizing process stops when it is not possible to find a connected path completely through white pixels.

In the resulting image, words are more compact and adapted to the subsequent operations (see Fig. 2). After re-



(a) Original image. (b) Resized image.

Figure 2. Illustration of the reduction of the image width.

sizing the image, the edges' weights of the graphs are estimated as explained next.

2.2.2 Design of the Weight Function

An immediate approach is to support the design of the weight function solely on the values of the incident nodes: if any of the corresponding pixels are black then a low cost is assigned to the edge; otherwise the edge assumes a high cost. We call this the `baseWeight` in Listing 2. However, the weight function can be generalized to account to other factors. To incorporate some prior knowledge about a text line into the shortest path process, we modified the weight function of the graph.

If a white pixel is part of a long horizontal run of pixels (an horizontal run is an horizontal sequence of consecutive pixels with the same value), it is more likely to be part of a inter-line space rather than a inter-character of inter-word space. Therefore, a term penalizing such pixels on long runs is included in the weight function. The pseudo-code for the weight function is provided in Listing 2, where `hRun1` and `hRun2` are the run-length of the runs containing the two pixels.

The quadratic penalizing term was experimentally selected over a set of training documents.

```
WeightFunction(pixelValue1, pixelValue2, hRun1, hRun2,
                                          NeighbourhoodType)
{
  value = min(pixelValue1, pixelValue2);
  hRun  = min (hRun1, hRun2);
  weight = baseWeight(value, NeighbourhoodType);
  if( max(pixelValue1, pixelValue2) == WHITE)
    weight = weight + (hRun /200)^2;
  return weight;
}
```

Listing 2: Pseudo-code for the weight Function. The base weight was set to 2 on black pixels and 6 on white pixels for 4-neighbourhoods and to 4 and 12 on for 8-neighbourhoods. For efficiency, weights were designed with integer values.

2.2.3 Main Cycle

The preprocessing is followed by the main cycle of the methodology, by successively finding the stable paths between the left and right margins, adding the paths found to the list of text lines, and erasing them from the image. Note that the algorithm computes all stable paths between margin points from one margin to the other, at each iteration. The erase operation sets to white the pixels on all connected components intersecting the detected text lines. The erase operation is necessary to ensure that a text line is not detected multiple times.

2.2.4 Stopping Rule

To stop the iterative text line search, a rule is used to validate the stable paths found; if none of them passes the checking, the iterative search is stopped. A path is discarded if it does not have a percentage of black pixels above a fixed threshold. The median percentage of blackness of all lines found in the first iteration of the main cycle provides the necessary reference (a threshold of 15% of the median value was empirically selected).

2.2.5 PostProcessing

After the main search step, valid text lines are post-processed. Although true text lines never intersect, the above algorithm may occasionally create intersecting lines, detected on different iterations. Local discontinuities can induce a stable path to zigzag up and down between consecutive text lines; on the next iteration, the detected path is likely to connect the remaining segments, and therefore intersect with the path detected in first place. To preclude such final, undesired state, lines are post-processed to remove intersections: for each image column, sort on y the pixels of the detected lines and assign the i-pixel to the i-line. After this simple process, lines may touch but they do not intersect.

Finally, lines are smoothed with a standard average low-pass filter. Considering a text line as a sequence $y(x)$ of y-

positions, a one-dimensional averaging filter is applied. A window size of $4\times$`interlinespace` was selected empirically. The baseline value `interlinespace` is estimated after the detection of the text lines, as the median y-distance between consecutive lines.

The text lines found so far are one-pixel thick through the text characters, but do not assign each and every character pixel to a line. Toward that end, we find the shortest path between the left and right margins, but *constrained* between two consecutive text lines. In this search, we favour paths through white pixels, trying to find the best separation-path between two consecutive text lines (see Fig. 3): a straightforward weight function is used in the graph, penalizing more black pixels (2) than white pixels (1). At the end, pixels between two consecutive separation-paths are assigned to the same text line.

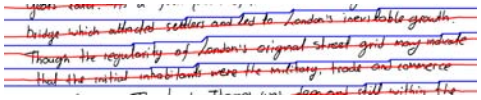


Figure 3. Detail of the postprocessing. The red paths are the stable paths found on the main cycle of the algorithm; the blue ones are the paths found on the postprocessing, separating consecutive text lines.

3. Results

The proposed methodology was tested on a modern set of 300 handwritten images made available on the ICDAR 2009 handwriting segmentation competition. None of the documents include any non-text elements (such as lines, drawings, pictures and logos) and are all written in several languages including English, French, German and Greek. Almost all documents have two or more adjacent text lines touching in several areas. Some of them have variable skew angles among text lines. Furthermore, there are document images having text lines with different skew directions as well as document images having text lines with converse skew angles along the same text line. The appearance of accents is common in Greek and French handwritten documents. All documents are written from different writers and in the majority of the documents the distance of adjacent text lines is very small leading to a highly dense text. For all images, we have the corresponding groundtruth made available for the competition (the total number of text lines is 6276). One hundred images were used for designing the model (the design encompassed tuning the weight function, namely the term penalizing long runs of white pixels, and the threshold of the stopping rule); the performance of the model was assessed on the other 200 images.

The assessment of the performance of the text line detection was based on the metric introduced in [5], with the evaluator's acceptance threshold *th* defined as 0.95. The performance evaluation is based on counting the number of matches between the areas detected by the algorithm and the areas in the ground truth. A table is contructed with entries calculated according to the overlap of the labeled pixel sets as either text lines or words and the ground truth. The detection rate (DR) and recognition accuracy (RA) are calculated from the number of one-to-one one-to-many matches extracted from the table. The proposed stable text line detection method achieved a detection rate of 97% and a recognition accuracy of 97%.

Figures 4 and 5 contain text line segmentation results of the proposed methodology. Figure 4 represents the worst result, while Figure 5 two of the images where lines were perfectly segmented. We see that text lines of variable length and with highly asymmetric character density still pose difficulties. Moreover, errors in detecting one line tend to propagate to other lines. Finally, it is worth mentioning the robustness of the method to skewed and curved text.

(a) Original image, with results from the main cycle and postprocessing superimposed.

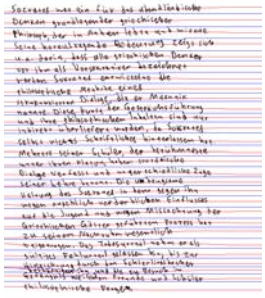
(b) Final segmentation.

Figure 4. Text line segmentation with poor results.

4. Conclusion

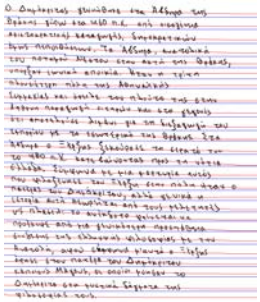
This paper presented a robust algorithm for the automatic detection of text lines in handwritten documents. The proposed method uses a very simple but fundamental principle to assist detection and avoid the difficulties typically posed by handwritten documents. The stable path approach for text line detection algorithm is adaptable to a wide range of image conditions, thanks to its intrinsic robustness to skewed images, discontinuities, and curved text lines. The fact that this same idea has been applied before to the detection of staff lines in music scores confirms the robustness and generality of the framework.

Our algorithm is to be further improved by refining the definition of the weight function (incorporating more knowledge about a text document in it) and the pre- and post-processing procedure. We also intend to evaluate on historical documents and on more languages, as in Chinese

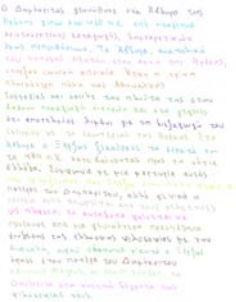


(a) Original image, with results from the main cycle and postprocessing superimposed.

(b) Final segmentation.



(c) Original image, with results from the main cycle and postprocessing superimposed.



(d) Final segmentation.

Figure 5. Text line segmentation with perfect results.

documents, via customizing the weight function and training with document images of specific languages.

Acknowledgments

This work was partially funded by Fundação para a Ciência e a Tecnologia (FCT) - Portugal through project PTDC/EIA/71225/2006.

References

[1] S. Avidan and A. Shamir. Seam carving for content-aware image resizing. *ACM Trans. Graph.*, 26(3):10, 2007.

[2] J. S. Cardoso, A. Capela, A. Rebelo, C. Guedes, and J. F. P. da Costa. Staff detection with stable paths (online). *IEEE Transactions Pattern Analysis Machine Intelligence*, 2009.

[3] Y. Li, Y. Zheng, D. Doermann, and S. Jaeger. Script-independent text line segmentation in freestyle handwritten documents. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30:1313–1329, 2008.

[4] L. Likforman-Sulem, A. Zahour, and B. Taconet. Text line segmentation of historical documents: a survey. *International Journal on Document Analysis and Recognition*, 9:123–138, 2007.

[5] G. Louloudis, B. Gatos, I.Pratikakis, and C. Halatsis. Text line and word segmentation of handwritten documents. *Pattern Recognition*, 2009.

[6] H. Oliveira and J. S. Cardoso. Image retargeting using stable paths. In *Proceedings of the International Conference on Computer Vision Theory and Applications*, pages 40–47, 2009.

[7] F. Yin and C.-L. Liu. Handwritten chinese text line segmentation by clustering with distance metric learning. *Pattern Recognition*, 2009.