Layout Analysis of Urdu Document Images

Faisal Shafait*, Adnan-ul-Hasan, Daniel Keysers*, and Thomas M. Breuel**
*Image Understanding and Pattern Recognition (IUPR) research group
German Research Center for Artificial Intelligence (DFKI GmbH)
D-67663 Kaiserslautern, Germany
**Department of Computer Science, Technical University of Kaiserslautern
D-67663 Kaiserslautern, Germany
faisal.shafait@dfki.de, adnan-ul-hasan@yahoo.com, daniel.keysers@dfki.de, tmb@informatik.uni-kl.de

Abstract— **Layout analysis is a key component of an OCR system. In this paper, we present a layout analysis system for extracting text-lines in reading order from Urdu document images. For this purpose, we evaluate an existing system for Roman script text on Urdu documents and describe its methods and the main changes necessary to adapt it to Urdu script. The main changes are: 1) the text-line model for Roman script is modified to adapt to Urdu text, 2) reading order of an Urdu document is defined. The method is applied to a collection of scanned Urdu documents from various books, magazines, and newspapers. The results show high text-line detection accuracy on scanned images of Urdu prose and poetry books and magazines. The algorithm also works reasonably well on newspaper images. We also identify directions for future work which may further improve the accuracy of the system.**

I. Introduction

Document image layout analysis is a crucial step in many applications related to document images, like text extraction using optical character recognition (OCR), reflowing documents, and layout-based document retrieval. Layout analysis is the process of identifying layout structures by analyzing page images. Layout structures can be physical (text, graphics, pictures, ...) or logical (title, author, abstract, ...). The identification of physical layout structures is called physical or geometric layout analysis, while assigning different logical roles to the detected regions is termed as logical layout analysis [1]. In this paper we are concerned with geometric layout analysis of Urdu documents. The task of a geometric layout analysis system is to segment the document image into homogeneous zones, each consisting of only one physical layout structure, and to identify their spatial relationship (e.g. reading order). In this paper, our contributions to Urdu document layout analysis are:

1) Giving an overview of the state-of-the-art in Urdu OCR
2) Applying a layout analysis system to Urdu documents that has proven to work well on Roman script.
3) Developing an Urdu text-line model, which is suitable for modeling Urdu text-lines in popular Urdu fonts.
4) Modifying the reading-order algorithm for Roman script documents to adapt to Urdu script.

Urdu is the national language of Pakistan with more than 140 million speakers. Urdu script belongs to the family of scripts based on Arabic script. It is a cursive script, i.e. individual characters are usually combined to form ligatures. Although many fonts are available for Urdu, the predominant fonts are *Nastaliq* and *Naskh*. In order to automatically convert an Urdu document image to electronic form, an Urdu OCR system is needed. However, there has been very little work in the area of Urdu OCR. Husain et al. [2] proposed an Urdu OCR system for the Nastaliq font based on neural networks. They skip the layout analysis step to concenterate more on the OCR part. Since the Urdu language has above 17,000 ligatures, they used only the frequent ones for training and testing. Pal et al. [3] present an approach for recognizing printed Urdu script. First, they perform skew correction of the document followed by line segmentation by horizontal projection. Then each extracted line is segmented into individual ligatures by vertical projection and connected component labeling. Finally, the basic characters and numerals are fed to the OCR system for recognition since they do not consider the recognition of compound characters. One of the main reasons for the lack of a complete Urdu OCR system is limited support of the Urdu language in computing. Many softwares do not have an Urdu user interface, and only a few Urdu editors are available. Recent projects like the *Urdu Localization Project* [4] aimed at improving the Urdu language support in computing environments. These advances in the support of Urdu in computing are leading to increased interest in the digitization of Urdu literature. A large project is being carried out to preserve the work of Pakistan's national poet, Allama Muhammad Iqbal [5].

In this paper, we focus on layout analysis of Urdu documents. Segmentation of a page image into individual lines by horizontal projection as used by Pal et al. [3] is a primitive approach and works only for clean, single-column documents. Over the last two decades, several layout analysis algorithms have been proposed in the literature (for a literature survey, please refer to [1], [6], [7]) that work for different layouts and are quite robust to the presence of noise in the document. Many of these algorithms have come to wide-spread use for analyzing document images in different scripts. Such algorithms include the X-Y cut [8], the smearing algorithm [9], whitespace analysis [10], Docstrum [11], and the Voronoi-diagram based approach [12]. A common feature of these algorithms is that they have many parameters that can be tuned to segment a particular type of documents. However, if the algorithm has to be

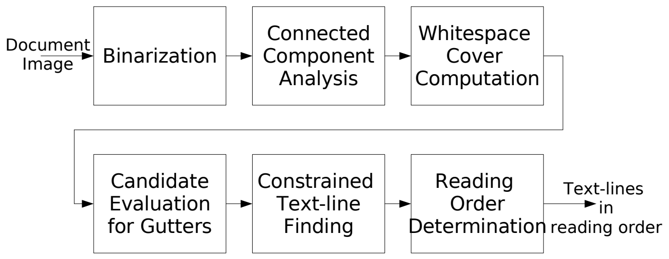

Document Image
Binarization
Connected Component Analysis
Whitespace Cover Computation
Candidate Evaluation for Gutters
Constrained Text-line Finding
Reading Order Determination
Text-lines in reading order

Fig. 1. Block diagram of the layout analysis system.

applied on a diverse collection of documents, the parameter tuning becomes a challenging task. Breuel [13] has proposed a layout analysis system using globally optimal geometric algorithms, combined with robust statistical models, to model and analyze the layouts of documents. This layout analysis system is applicable to a wide variety of languages and layouts, and is robust to the presence of noise and spurious features in a page image. A particular advantage of this system is that it is nearly parameter-free. Hence we focus on adapting the layout analysis system described in [13] to Urdu script documents.

II. Urdu Document Analysis

The document layout analysis system in [13] consists of the following main steps:

1) Find empty whitespace rectangles that completely cover the page background.
2) The whitespace rectangles are evaluated as candidates for column separators or gutters based on their aspect ratio, width, and proximity to text-sized connected components.
3) Find text-lines that respect the columnar structure of the document.
4) Determine the reading order of the text-lines using constraints on the geometric arrangement of text-line segments on the page.

Since the algorithm takes the connected components of a document image as input, we include additional document preprocessing steps for binarization and connected component analysis into our complete layout analysis system. A block diagram of the system is shown in Figure 1. A brief description of each step is explained in the following subsections.

A. Document Binarization

In order to binarize a given greyscale document image, we use a local adaptive thresholding technique [14]. Consider the gray value of a pixel at position (x, y) in the image be represented by $g(x, y)$. First, the local region of the pixel is defined by a $w \times w$ window centered at the position of the pixel. Then minimum and maximum gray value $g_{\min}$ and $g_{\max}$ in the local region of the pixel are computed. Then a value for the local threshold is chosen based on the difference between $g_{\max}$ and $g_{\min}$ as defined in Equation 1.

$$t(x,y) = \begin{cases} (g_{\max} + g_{\min})/2 & \text{if } g_{\max} - g_{\min} \geq T \\ g_{\max} - (T/2) & \text{otherwise} \end{cases} \tag{1}$$

The threshold T allows local variations within the background or foreground pixels. We chose $w = 40$ and $T = 50$ for our experiments. An example image binarized with using these parameter values is shown in Figure 2.

B. Connected Component Analysis

After binarization, the next step is to extract connected components from the document image. We use a fast labeling approach to extract connected components as described in [15]. The algorithm is based on a union-find data structure and returns the bounding boxes of the connected components. The bounding boxes overlayed on the original image are shown in Figure 2. Noise removal is done at this stage by rejecting very small and very large connected components. If print-through from the other side of a double-sided printed document is also present in the image, it appears as isolated dots and speckles as a result of binarization. Such dots and speckles are also filtered out at this stage.

C. Whitespace Cover Computation

The whitespace analysis algorithm described by Breuel [16] analyzes the structure of the white background in document images. The target is to find a set of maximal white rectangles (called *covers*) whose union completely covers the page background, and none of them overlaps with any of the connected components on the page. The key idea behind the algorithm is similar to quicksort or branch-and-bound methods. The details of the algorithm can be found in [16]. The algorithm returns globally optimal covers in decreasing order with respect to their area, until a minimum area is reached or the maximum specified number of rectangles have been obtained. Usually 300 rectangles are sufficient to completely cover the page background.

D. Evaluation of Candidates for Column Separators

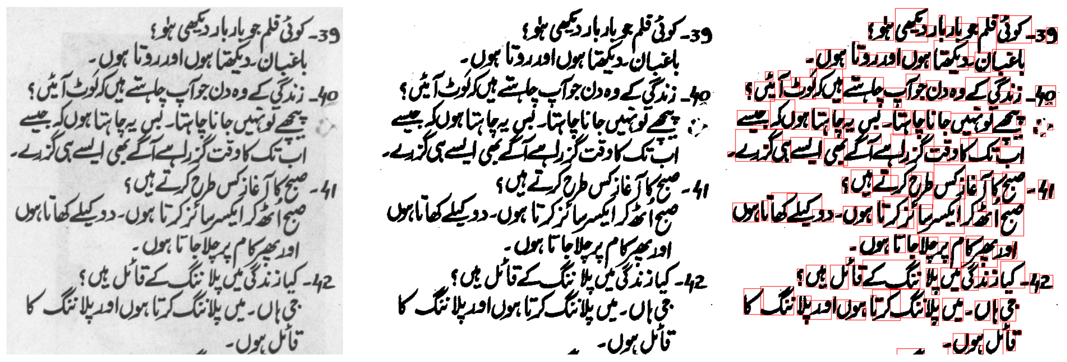
The whitespace rectangles are evaluated as candidates for column separators based on the following constraints:

1) Column-separating rectangles must have an aspect ratio of at least 1:3
2) Column-separating rectangles must have a width of at least 1.5 times of the mode of the distribution of widths of inter-word spaces.
3) Column-separating rectangles must be adjacent to at least four character-sized connected components on their left or their right side.

The candidates satisfying these criterion are selected as column-separators. Some example of column-separators found in this way are shown in Figure 5.

E. Constrained Text-line Finding

The idea behind the constrained text-line finding algorithm is to take the column boundaries identified by the background analysis described in the previous subsection and introduce



(a) Original image

(b) Binarized image

(c) Bounding boxes of the connected components

Fig. 2. Example image illustrating the binarization and connected component analysis steps. Note that in the original image, besides noise, print-through from the other side of the page also appears. The binarization successfully removes most of the print-through. Some parts of the print-through may appear as isolated dots and speckles in the binarized image. These dots are removed along with other noisy components during the connected component analysis. The bounding boxes of the extracted connected components after noise removal are overlayed on the binary image to obtain image on the left.

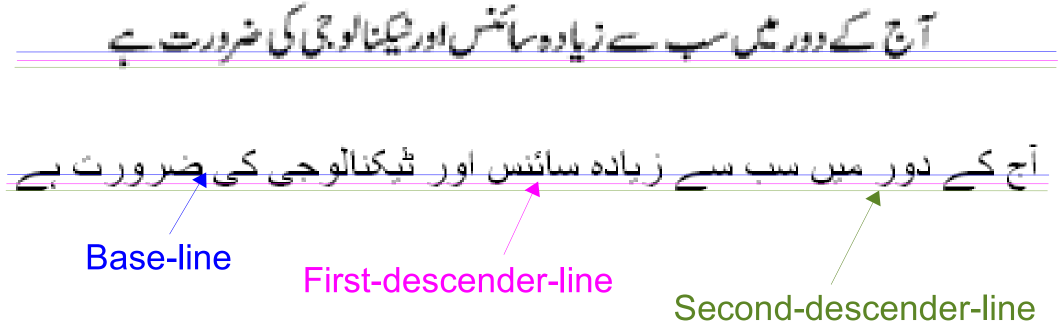

آج کے دور میں سب سے زیادہ سائنس اور ٹیکنالوجی کی ضرورت ہے
آج کے دور میں سب سے زیادہ سائنس اور ٹیکنالوجی کی ضرورت ہے
Base-line
First-descender-line
Second-descender-line

Fig. 3. Urdu text-line modeling using two descender lines. The text-line at the top shows a piece of text in Nastaliq font, whereas the text-line at the bottom shows the same piece of text in Naskh font. It is interesting to note here that although different characters appear at different heights in both fonts, the resulting text-lines have two descender lines.

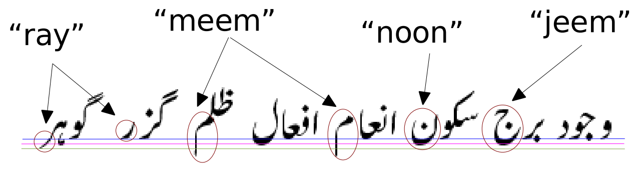


Fig. 4. An example image showing the descender distance of different characters in the Nastaliq font. The characters like “jeem”, “noon” etc. make a line of descenders whether they appear alone or appear at the end of a compound character. On the other hand, characters like “meem” descend lower than the line of descenders formed by characters mentioned previously. Also, characters like “ray” may either lie on the baseline, on first descender line, or on the second descender line depending on their position in the compound character, and other characters making the compound character.

them as “obstacles” into a statistically robust, least square, globally optimal text-line detection algorithm. The details of the algorithm can be found in [16]. The algorithm uses a text-line model based on parameters r, θ, d, where r is the distance of the baseline from the origin, θ is the angle of the baseline from the horizontal axis, and d is the distance of the line of descenders from the baseline.

This text-line model was developed for Roman script documents. For Urdu script the descender distance of all compound and individual characters is not the same as illustrated in Figure 4. Another source of variation comes from the font, for example the character “bari yay” (the last character of the textline shown in Figure 3) lies on the baseline in the Nastaliq font, whereas it lies on the second descender line in Naskh font. Hence the single-descender model used for Roman script is not suitable for modeling text-lines in the Urdu script. Therefore, we develop a new text-line model for Urdu script using two descenders as shown in Figure 3. These lines are parameterized by parameters (r, θ, d_1, d_2). The RAST algorithm [17] is used to find globally optimal text-lines using this parameter set.

The text-lines returned by the RAST algorithm are verified by using image statistics about average width and height of a character. This works very well for Roman script documents, where the x-height (height of a lower case letter ’x’) gives a very reliable estimate of the dominant font size. However, for Urdu script, the presence of a large number of dots and diacritics make it difficult to estimate the average character height reliable. The presence of compound characters further complicates the situation. Hence, the results obtained on Urdu script are not as robust as those in the Roman script. This point will be further discussed in Section III.

TABLE I

Different types of errors made by the algorithm on each category of the dataset.

Document Type	Total Number of Text-lines	Correctly Detected Text-lines (%)	Over-Segmented Text-lines (%)	Under-Segmented Text-lines (%)	Missed Text-lines (%)	False Alarms (%)
Book	234	91.45	4.27	0.00	4.27	0.43
Poetry	286	92.31	4.55	0.00	3.15	0.00
Digest	702	80.63	11.54	0.00	7.84	0.25
Magazine	1158	90.07	4.14	0.86	4.75	11.83
Newspaper	819	72.16	7.81	4.15	15.87	16.36

F. *Reading Order Determination*

Reading direction of Urdu is from right to left, which is opposite to the Roman script which is read from left to right. Therefore we modify the reading order algorithm in [13] to obtain reading order of Urdu text-lines as follows:

1) Line segment a comes before line segment b if their ranges of x-coordinates overlap and if line segment a is above line segment b on the page.
2) Line segment a comes before line segment b if a is entirely to the *right* of b and if there does not exist a line segment c whose y-coordinates are between a and b and whose range of x-coordinates overlaps both a and b.

Applying these two ordering criteria are sufficient to define partial order of text-line segments. This partial order is then extended to a total order of all elements using a topological sorting algorithm [18].

III. Experiments, Results, and Discussion

In order to evaluate the performance of the described layout analysis system, we collected 25 images of Urdu text from different sources. The images are categorized into five classes: *book*, *poetry*, *digest*, *magazine*, and *newspaper*. There are 5 images of each class in the dataset. The dataset is made publicly available and can be downloaded freely from [19]. Although the dataset is quite small in size, it contains many types of layouts. Hence it can be used to evaluate the performance of a layout analysis algorithm for Urdu documents. However, for a thorough evaluation, a larger dataset may be required. Example images showing the result of the algorithm are shown in Figure 5.

The evaluation of the algorithm is done in two parts. The first part analyzes the errors made in text-line detection, and the second part evaluated the reading order algorithm.

A. *Text-line Detection Accuracy*

The text-line detection accuracy measures how many text-lines were correctly detected. A text-line is said to be **correctly detected** if it does not fall into any of the four types of error defined below.

Oversegmented text-lines: the number of text-lines that are either split into more than one text-line, or partially detected.

Undersegmented text-lines: the number of text-lines merged with some other text-line.

Missed text-lines: the number of text-lines that were not found by the algorithm.

False alarms: the number of detecting text-lines originating from noise or non-text elements.

These error types are similar to those used for determining text-line detection accuracy in Roman script documents [20], [21].

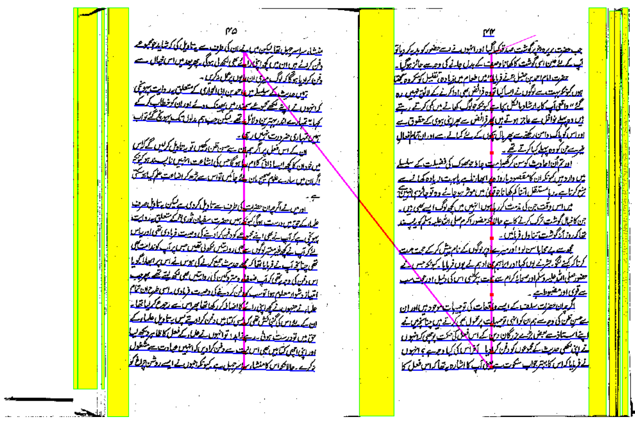
Table I summarizes the results obtained by applying the algorithm to the test data, and manually inspecting the resulting text-lines detected by the system. Based on these results, the following observations can be made:

- The algorithm works quite well on *book*, *magazine*, and *poetry* layouts. A few number of missed error are those in which the text-line consisted of only one connected component. The text-line detection algorithm needs at least two components to make a line. The oversegmentation errors occur due to the page curl.
- In the *digest* layout, many oversegmentation errors occur due to enumerated lists, in which the enumerator-symbol is segmented separately.
- In the *newspaper* layout, a lot of text-lines are missed. This is due to very small inter-line spacing between two consecutive text-line. Due to this small spacing connected components from two neighboring text-lines sometimes merge together. In such a case the upper text-line is missed. Another source of missed error was the presence of inverted text resulting in poor binarization of the image as shown in Figure 6. Also, a number of false alarms appear in these layouts due to non-text elements on the page.

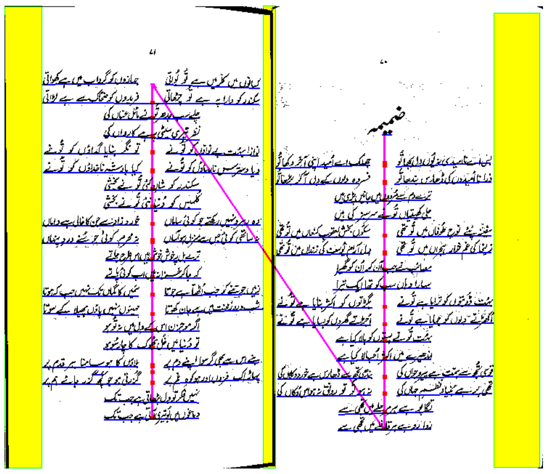
B. *Reading Order Determination Accuracy*

The reading order determination algorithm heavily depends on the text-line detection accuracy. Manual inspection of the results showed the following types of errors:

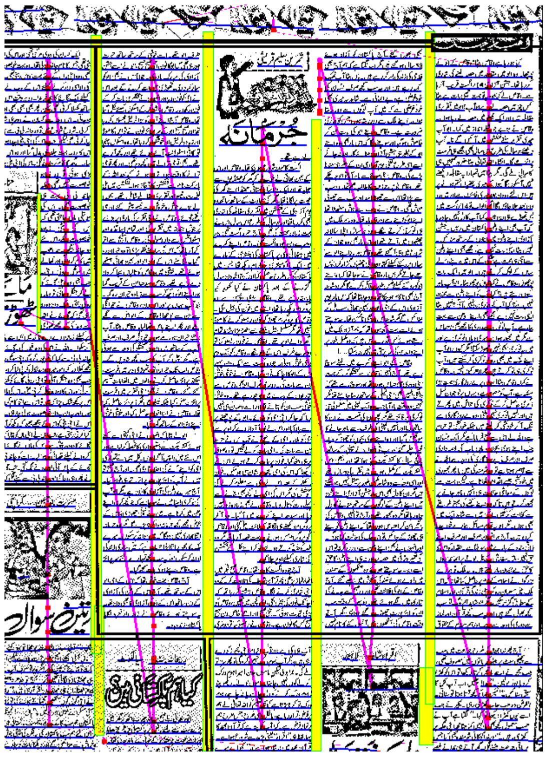
1) If two text-lines from different text columns are merged, they are interpreted as a separator and hence the algorithm gives the wrong reading order.
2) In some cases, the separation between different sections of a multi-column document is not represented by a text-line spanning along the columns, but a ruling (thick horizontal black line) is used instead. In that case the algorithm does not detect the start of a new section.
3) The reading order of poetry in two-column layout is not handled by the algorithm.



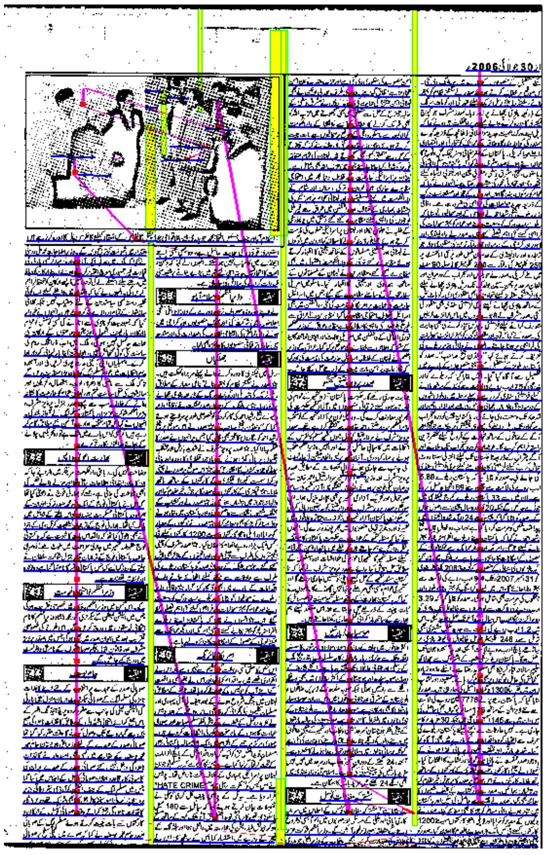
(a) Facing pages of a prose book



(b) Facing pages of a poetry book



(c) Page from a magazine



(d) Page from a newspaper

Fig. 5. Some example images illustrating results of the proposed layout analysis system. The yellow rectangles show the detected vertical gutters. Thin horizontal blue lines indicate detected text-line segments, and the thick magenta lines running down and diagonally across the image indicate reading order. Note that images and graphics were not removed, so they result in some spurious text-lines.

C. Problems and Future Directions

Based on the experimental results obtained in this paper, the layout analysis system described in this paper can possibly be improved in the following ways:

1) Image statistics like x-height (height of a lower case letter 'x') estimation play an important role in automatically adjusting parameters of the algorithm to the resolution and font size of the input document. Incorporating such a statistical analysis for Urdu documents may reduce the number of missed lines.
2) Using a binarization technique that can handle both normal and inverted text on the same image will result in improved performance on *newspaper* images.
3) A large dataset of scanned Urdu documents with ground-truth should be collected so that the results are less affected by the contents of an individual image.

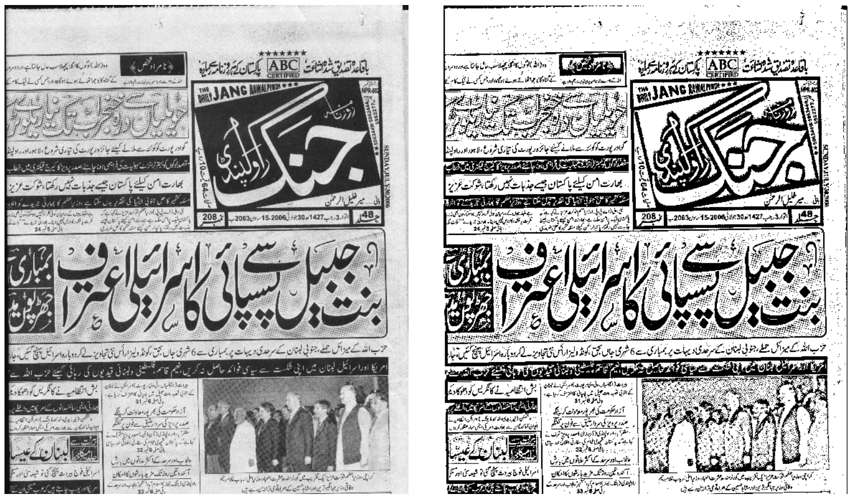


Fig. 6. An example image cropped from the front page of a newspaper containing both normal and inverted text in the same image. The binarization step fails to identify the inverted text and makes it a part of the page background (white) as shown in the binarized image.

4) If the skew of the scanned document is so large that axis-aligned gutters cannot be found, an algorithm to extract whitespace rectangles at arbitrary orientations [22] can be used.
5) The reading order algorithm cannot handle poetry written in two column format, as it is misinterpreted as a two-column text. Hence a different algorithm must be used for poetry documents in two-column format.

IV. Conclusion

In this paper we presented a layout analysis system for Urdu documents. This system had shown to work well on Roman script so it was adapted to Urdu documents. The described system was tested on 25 scanned images from different sources like books, magazines, and newspapers. The algorithm achieved an accuracy of above 90% in terms of text-line detection for books and magazine images. The accuracy decreased to about 80% for documents from digest due to small inter-line spacing and presence of enumerated lists. Newspaper documents proved to be the toughest class presenting several challenges like many font sizes within the same image, small inter-line spacing, inverted text, and poor quality of page resulting in lot of noise. The text-line detection accuracy for the newspaper images was 72%.

Acknowledgments

This work was partially funded by the BMBF (German Federal Ministry of Education and Research), project IPeT (01 IW D03).

References

[1] R. Cattoni, T. Coianiz, S. Messelodi, and C. M. Modena. Geometric layout analysis techniques for document image understanding: a review. Technical Report 9703-09, IRST, Trento, Italy, 1998.

[2] S. A. Husain and S. H. Amin. A multi-tier holistic approach for urdu nastaliq recognition. In *IEEE Int. Multi-topic Conference*, Karachi, Pakistan, Dec. 2002.

[3] U. Pal and A. Sarkar. Recognition of printed Urdu script. In *Seventh Int. Conf. on Document Analysis and Recognition*, pages 1183–1187, Edinburgh, UK, Aug. 2003.

[4] S. Hussain. Urdu localization project. In *Workshop on Computational Approaches to Arabic Script-based Languages*, Geneva, Switzerland, 2004.

[5] http://www.allamaiqbal.com/.

[6] G. Nagy. Twenty years of document image analysis in PAMI. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(1):38–62, 2000.

[7] S. Mao, A. Rosenfeld, and T. Kanungo. Document structure analysis algorithms: a literature survey. *Proc. SPIE Electronic Imaging*, 5010:197–207, Jan. 2003.

[8] G. Nagy, S. Seth, and M. Viswanathan. A prototype document image analysis system for technical journals. *Computer*, 7(25):10–22, 1992.

[9] K. Y. Wong, R. G. Casey, and F. M. Wahl. Document analysis system. *IBM Journal of Research and Development*, 26(6):647–656, 1982.

[10] H. S. Baird. Background structure in document images. In H. Bunke, P. Wang, and H. S. Baird, editors, *Document Image Analysis*, pages 17–34. World Scientific, Singapore, 1994.

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

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

[13] T. M. Breuel. High performance document layout analysis. In *Symposium on Document Image Understanding Technology*, Greenbelt, MD, April 2003.

[14] E. R. Davies. *Machine Vision: Theory, Algorithms, Practicalities, (Second Edition)*. Academic Press, San Diego, 1997.

[15] Berthold K. P. Horn. *Robot Vision*. MIT Press, Cambridge, MA, 2001.

[16] T. M. Breuel. Two geometric algorithms for layout analysis. In *Document Analysis Systems*, pages 188–199, Princeton, NY, Aug. 2002.

[17] T.M. Breuel. Recognition by Adaptive Subdivision of Transformation Space: practical experiences and comparison with the Hough transform. In *IEE Colloquium on 'Hough Transforms' (Digest No.106)*, pages 71–74, 1993.

[18] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms, (Second Edition)*. MIT Press, Cambridge, MA, 2001.

[19] http://www.iupr.org/demos_downloads/.

[20] J. Liang, I. T. Phillips, and R. M. Haralick. Performance evaluation of document structure extraction algorithms. *Computer Vision and Image Understanding*, 84:144–159, 2001.

[21] F. Shafait, D. Keysers, and T. M. Breuel. Pixel-accurate representation and evaluation of page segmentation in document images. In *18th Int. Conf. on Pattern Recognition*, pages 872–875, Hong Kong, China, Aug. 2006.

[22] T. M. Breuel. An algorithm for finding maximal whitespace rectangles at arbitrary orientations for document layout analysis. In *Seventh Int. Conf. on Document Analysis and Recognition*, pages 66–70, Edinburgh, UK, Aug. 2003.