Segmentation of Heterogeneous Documents into Homogeneous Components using Morphological Operations

Nasid Habib Barna
Department of Computer Science and Engineering
University of Dhaka
Bangladesh
barna.bd96@gmail.com

Tisa Islam Erana
Department of Computer Science and Engineering
University of Dhaka
Bangladesh
tisaislamerana@gmail.com

Shabbir Ahmed
Department of Computer Science and Engineering
University of Dhaka
Bangladesh
shabbir@cse.du.ac.bd

Hasnain Heickal
Department of Computer Science and Engineering
University of Dhaka
Bangladesh
hasnain@cse.du.ac.bd

***Abstract*—The research on document layout analysis has been widespread over a large arena recently and is craving for more efficiency day by day. Document segmentation is an important preprocessing step before analyzing the layouts. This paper presents a language-independent document segmentation system that segments a heterogeneous printed document into homogeneous components like halftones and graphics, texts and tables including its individual cells. From an input document page homogeneous components are segmented in three steps with three separate modules, which are- extraction of halftone images, extraction of tables and segmentation of text blocks. These modules altogether build the whole page segmentation system which takes an input image of heterogeneous document page and produces an output with explicitly indicated homogeneous segments with colored bounding boxes. The modules use morphological operations to detect the components. To improve the performance of image segmentation Residual Image Fragments Retrieval (RIFR) is proposed. The paper also proposes Text Extraction from Table Cells (TETC). Combining RIFR and TETC together we get an overall accuracy of 93%. Table and cell detection have a higher accuracy of 96% whereas image and texts have around 90% accuracy.**

***Index Terms*—Document Segmentation, Morphological Operation, Text Extraction, Table Extraction, Image Extraction, Language Independent Page Segmentation System**

I. Introduction

With the progress of technology, there are absolute demands for storing all documents in the virtual storage in digitized form. On this note, Optical character recognition (OCR) is the process of electronic conversion of images of printed, typed or handwritten text into digital format. These document pages appear to be heterogeneous, that is different types of components are scatteredly positioned in the document. Before feeding texts of these heterogeneous document to an OCR, the document must be segmented so that different components such as texts, halftones, graphics, tables and symbols are separated and only text components are thus identified. Precision of the segmentation directly affects the accuracy of an OCR module. Also detection and labeling of different regions in a document would help in great extent while storing printed documents in the virtual storage in digitized form in their correct reading order. Therefore document segmentation is a important preprocessing step for automatic document analysis.

S Eskenazi et al. [1] performed an extensive survey on a number of approaches that have been proposed for document image segmentation. Among all the previous works in document page segmentation, very few system is found to be working efficiently as a language independent system which works on image, text and table separation simultaneously. Table is a special component in documents because it is comprised of many cells which contain texts. K. Sobottka et al. [2] used a marginal feature- the difference of color to detect text region. So text with different font style,font size etc might not get identified as texts. Anil K.Jain et al. [3] proposed a texture-based page segmentation algorithm which uses classifier for different components in an image which are text, line-drawing, halftone and background. Text and line-drawing regions are further separated using connected component analysis. They only considered image and text in the documents. T Kasar et al. [4] proposed a method to detect tables by identifying the vertical and horizontal line separators and their properties. They extracted 26 low-level features and used SVM classifier to check whether the line belongs to a table or not. Dan S. Bloomberg et al. [5] and Syed Saqib Bukhari et al. [6] proposed and implemented algorithms based on multiresolution morphology that segments documents into texts and halftone images. They also only considered image and text in documents where some fragments of images parts were recognized as texts. Vikas J Dongre et al. [7] used histogram approach to split this document into lines, words and characters. But this produces erroneous result due to their top to bottom separation approach. Here many unconnected vertical lines were recognized as separate symbols. Youbao Tang et al. [8] proposed matched filtering and top-down grouping approach for text line segmentation in handwritten documents. They did not consider image or table in the documents.

This paper proposes a language independent document segmentation system which is used for segmenting a hetero-

978-1-5386-5892-5/18/$31.00 ©2018 IEEE
ICIS 2018, June 6-8, 2018, Singapore

geneous document into homogeneous components working on image, text and table separation at the same time. The whole system is divided into three modules which are used for segmenting three major components of any document page- Images, Texts and Tables. First the input document goes through the image segmentation module and separates the individual images from the text regions of the document. Proposed RIFR is used to improve the performance of this segmentation. Then the text region which may contain tables is passed through the table separation module and as a result the tables are separated from the text regions. These tables further go through proposed TETC module for detecting individual cells and cropping these cells into individual images. The rest of the document contains texts possibly arranged into some paragraphs. These paragraphs are identified as texts and separated as text blocks. Thus the whole document which was heterogeneous, consisting different components, is segmented into three type of homogeneous components: Images, Texts and Tables.

We approached in a way to deal with the shortcomings of the previous solutions [2] [5] [7] [3]. We implemented parts of Bloomberg's Algorithm [5] for image segmentation and proposed Residual Image Fragments Retrieval (RIFR) for improving the performance. Text Extraction from Table Cells (TETC) is proposed to extract table cells. The overall accuracy of the system is 93%. For image, text, table and cell extraction, the accuracy rate is 90.7%, 90%, 96.8% and 95% respectively.

II. Proposed Method

This section describes the proposed approach for segmentation of a heterogeneous document into homogeneous components. Segmentation is performed in three steps:

- Images are separated as halftone components from the text part of the input image. We implemented the image separation technique using the idea of Bloomberg's algorithm [5]. Then we used the proposed algorithm RIFR to improve the performance.
- After separating images, tables ,being part of the text regions, are separated from actual text regions. Proposed TETC algorithm is used to extract individual cells.
- Connected text components are brought together into one block and thus different text regions are segmented into homogeneous blocks.

A. Image Preprocessing

In the aim of generating suitable input data for proper segmentation, we first went through a denoising method to lower the noise to a reasonable level. Angle correction is also needed to ensure accuracy of the segmentation process. And to do so, at first all the coordinates that are part of the foreground is found. Then the minimum rotated rectangle that holds the overall text area is found using those coordinates. Thus we get the angle values. The inverse of this value is the text skew angle. Then using the midpoint coordinates and the rotation angle the process of transformation, that is the angle correction, is done.

B. Halftone Image Separation

The output image obtained from the preprocessing module is next fed through the halftone image separator module with the intent to segment image regions from text regions. The algorithm is based on morphological techniques with the basic morphological operations and multiresolution morphology mentioned in paper [5]. Figure 1 demonstrates the halftone image separation operation.

Reducing the Image:
The preprocessed image after converting into binary is reduced four times than the initial size. This reduction is done with multiresolution morphological operation called *Threshold Reduction* introduced in paper [5].

- The image is subsampled into 16 x 1 dimension by two threshold reductions with threshold equal to one. Due to threshold being one, the action mimics dilation followed by subsampling which will solidify the halftone regions and create a subsampled image with 2^{n-2} pixels from 2^n.
- After that with a threshold equal to four, again the image is reduced. Since threshold is four, the text blobs will vanish mostly like an erosion with structuring element 2 x 2 followed by subsampling.
- Lastly with a threshold equal to three, the image is reduced again. After this the size of the image will reduce four times and halftone parts are consolidated with the intention of removing text parts.

Opening the Image: After threshold reduction an Opening operation with a structuring element 5 x 5 is done on the subsampled image so that the remaining text parts are removed and some parts of halftone components are preserved. This will generate the **seed image**.

Expanding the Image: Two 1 x 4 expansions are done on the seed image which will become equal in size with the 16 x 1 subsampled image.

Union of Overlapping Components: The expanded seed image is compared with the 16 x 1 subsampled image with the intention to unite the overlapping components.

Dilating the Image: To solidify the halftone mask, it is dilated with a structuring element 3 x 3.

Expanding the Image: The 16 x 1 subsampled mask returns to the size of the original image by performing two 1 x 4 expansions. This is the halftone mask image.

Residual Image Fragments Retrieval (RIFR): After separating the halftone mask image, it is removed from the original image which now consists of non-image parts. But since the process has gone through several reductions, the size of the halftone mask image may be smaller than the original halftone image. Thus fragments of halftone image are left behind in the original image. To retrieve these residual fragments from the text portion of the document, the mask is compared against the original image and the proposed RIFR is performed. The algorithm can be found in algorithm 1.

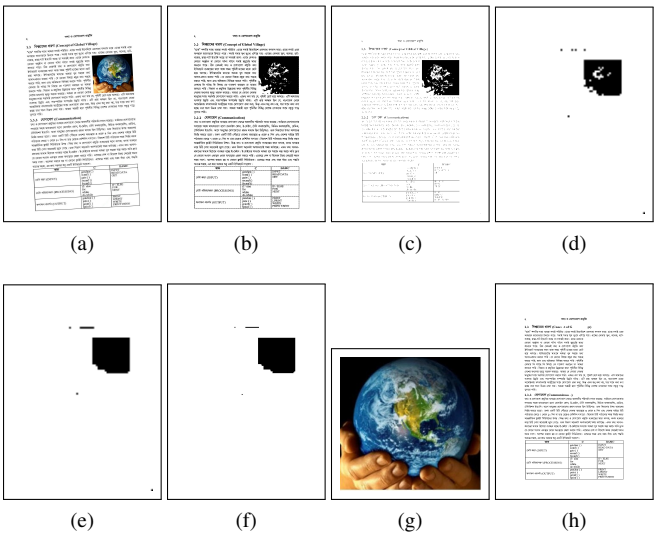


Fig. 1. (a) Input image (b) Rotated Binary Image (c) Reduced Image for Threshold One (d) Reduced Image for Threshold Four (e) Seed Image (f) Halftone Mask Image (g) Separated Image (h) Separated Non-image Part.

C. Table Separation

Two image outputs, one with only the image components and another with non-image components, were obtained from the previous module. The non-image part is then fed into the Table Separator module to detect and separate tables as we showed in figure 2. After table separation, individual cells in the table are detected.

Detection of Horizontal and Vertical Lines: Two clones of the main image are generated for separately detecting horizontal and vertical lines.

- **Horizontal Lines:** A horizontal structuring element is required for the horizontal lines to be detected. The clone image is eroded, dilated and closed with that structuring element and thus identified only the horizontal lines.
- **Vertical Lines:** Similar action was done for detecting vertical lines where a vertical structuring element is used.

As a result of erosion, only the lines fitting the structuring element retained in the image with a reduced length. After that dilation, the original size of the lines will be restored. Closing the image will fill the gaps in each line. Thus horizontal and vertical lines are separately detected in two images.

Line Uniforming: The lines that the table in the image is comprised of may differ in thickness. So we performed a manual operation to make all the lines of uniform thickness on both the detected horizontal and vertical lines. The steps are:

- Each of the lines is detected with endpoint coordinates with the help of *Houghline Transform*.
- Multiple lines that are in the proximity of a given threshold are considered to be within the current thickness of that particular line and replaced by a single line to make all lines of uniform thickness.

Algorithm 1 Residual Image Fragments Retrieval (RIFR)

Define: I(R,C) - Input image with R rows and C columns
O - Output image
range - Range of adjacent pixels

procedure RIFR (I, x, y, O)

```
for each pixel(x,y) in I do
    visited[pixel] = false
end for
range = 3
if visited[pixel] = true then
    return
end if
Q = ∅
ENQUEUE(Q, pixel(x,y) )
while Q ≠∅ do
    P1 = DEQUEUE (Q)
    if visited[Pl] = true then
        continue
    end if
    visited[Pl] = true
    O[P1] = 1
    for P2 = P1-range to P1+range do
        if I[P2] = 0 and visited[P2] = 0 then
            ENQUEUE(Q,P2)
        end if
    end for
end while
return
```

procedure MaskExtraction (Original, Mask)

```
for i = 0 to Rows in Mask do
    for j = 0 to Columns in Mask do
        if Original[i][j] = Mask[i][j] and Original[i][j] = 0
        then
            RIFR(Original, i, j, Output)
        else
            Output[i][j] = Original[i][j]
        end if
    end for
end for
return
```

- Adjacent lines that are separated by a minimum gap are joined together to make one.

Generating Table Mask: The table mask is generated by the union of the two images which are comprised of uniform horizontal and vertical lines respectively.

Cell Detection: For detecting every cell in each table we found the external and internal contours. The external contours represent the tables and the internals represent the cells in each table. Then the bounding rectangle of each contour is found which in turn gives the coordinates of each cells. From these coordinates Text Extraction from Table Cells is performed shown in algorithm 2. Now that we have the values of all the necessary coordinates, we then crop each table and their

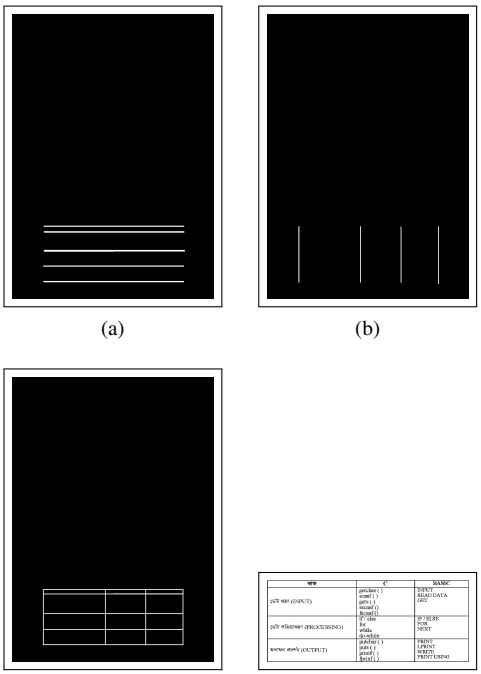


Fig. 2. (a) Separated Horizontal Lines. (b) Separated Vertical Lines. (c) Table Mask. (d) Cropped Table.

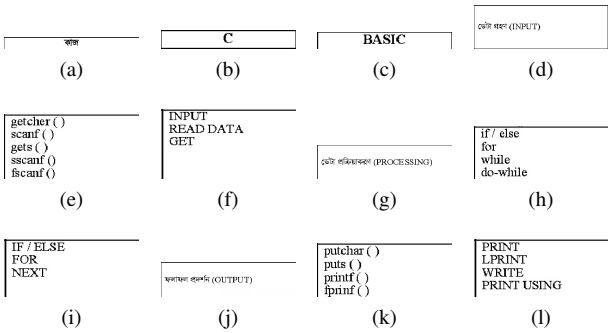


Fig. 3. Individual Cells of The Table

cells from the original picture as shown in figure 3.

D. Text Block Segmentation

As a result from the previous steps we get an image which contains only the texts. But this image could have texts in different blocks. So to split these blocks into homogeneous ones we further added some more steps.

Dilating the Image: In dilation, a pixel element becomes 1 if at least one pixel under the kernel is 1. This results in increasing the size of the foreground object. We dilated the image with a good number of iteration to get an image with a number of connected blocks. It can be shown as in figure 4b, the texts that are in the same block were connected with each other.

Contours: As connected components are thought to be in the same block, after dilating the image we have to find the

Algorithm 2 Text Extraction from Table Cells (TETC)

Define: H - Image containing horizontal lines
V - Image containing vertical lines
A - Approximate maximum area of a table cell.
ImageArray - Array of Image with tables removed.
Coordinates - List of coordinates of table and table cells.

procedure TETC (H, V, ImageArray)
TableMask = H + V
contours = findContours(TableMask)
j = 0
for each cnt in contours **do**
  x, y, w, h = boundingRect(cnt)
  area = contourArea(cnt)
  if area $>$ A **then**
    j = j + 1
    for p = y to y+h **do**
      for q = x to x+w **do**
        ImageArray[p][q] = 0
      end for
    end for
  end if
  Append x, y, x+w, y+h and j in Coordinates
end for
return Coordinates

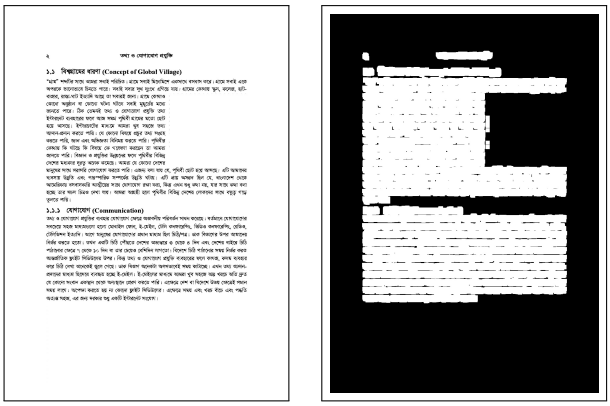


Fig. 4. (a) Input Image Containing Only Text. (b) Dilated Image

contours, an outline bounding the shape of each block. We stored the coordinates of boundary box of each object in an array. Then we found the minimal bounding rectangle for each specified point set. Thus we got different rectangles for each blocks. After this we cropped each rectangle from the main image and got separate images of each block as shown in figure 5.

After all the segmentation process is done, we would have all the different homogeneous segments in separate images. To identify which part goes where in the main image, all these homogeneous regions are then enclosed by bounding boxes of different color for three different parts of the input, that is- image, table and text.

(a) (b) (c) (d) (e) (f)

Fig. 5. (a) Input Image. (b), (c), (d), (e), (f) Segmented text blocks

III. Experimental Result and Discussion

To test the proposed approach for segmentation of a heterogeneous page of a book into homogeneous segments described in section II we performed a profound experiment. This section focuses on the performance analysis of our proposed method which can be divided into several categories-

A. Experimental Setup

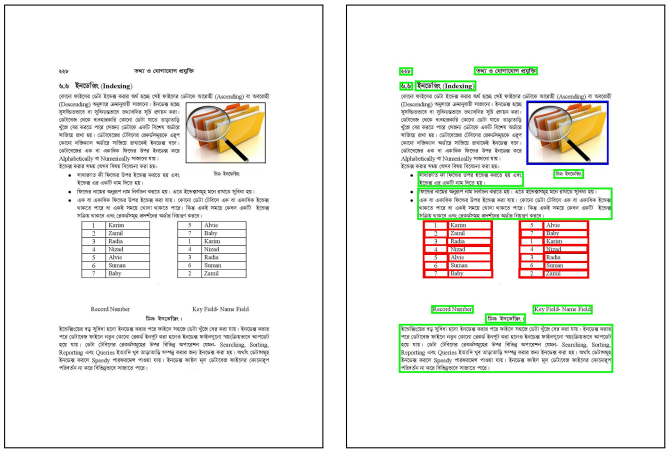
We have run our experiments on a computer with 2.50 GHz Intel Core i5 processor with 8 GB RAM, running Windows 7 Ultimate 64 bit operating system. The experiment was written mainly in Python using PyCharm 2017.2. The major component used in this experiment is OpenCV [9].

B. Implementation and Experimental Result

The major parts of implementation of the experiment are-

- Halftone Image Segmentation Module.
- Table Extraction Module.
- Text Block Segmentation Module.

To conduct the experiment properly we collected a huge amount of data. These data are images extracted from documents which are categorized into three types namely **Digitally Generated PDF**, **Scanned PDF from Online Sources** and **Manually Scanned Document**. We used Epson L220 scanner to scan the paper documents. The collected pdf books' pages were converted to images using a pdf to image converter. We used OpenCV's built in function *fastNlMeansDenoisingColored* for noise reduction. With the determined skew angle value and the midpoint coordinates we deskewed the image using openCV functions *getRotationMatrix2D* and *warpAffine*. After these preprocessing steps we implemented the segmentation algorithm which has been described in section II.



(a) (b)

Fig. 6. Segmentation of an image into homogeneous different regions marked by colored bounding boxes.

Result: The bounding boxes represents the homogeneous segments of a heterogeneous page. Three different colors are used for representing three categories of homogeneous regions. Blue, Red and Green colored bounding boxes are used for Images, Tables and Text Blocks respectively.

- Images
- Tables
- Text Blocks

Figure 6 shows our system's output where images, tables and text blocks are identified and marked with different colored bounding boxes.

C. Performance Analysis

We performed an extensive experiment for observing and analyzing the result of the implementation of our proposed method described in section II. We performed the experiment manually with a data-set of 100 images and analyzed the results based on detection accuracy.

Whether homogeneous components segmented by different segmentation modules are detected correctly is a big concern. It may happen that the Image Separator module detected some part of the text as an image, or some image part are left behind in the non-image portion and later is segmented as a text block. So the confusion matrix was formed and from the matrix we calculated *True Positive Rate (TPR)* and *False Positive Rate (FPR)* for each of the modules and the result is shown below with confusion matrix and column chart.

Confusion Matrix:

Table I represents the confusion matrix for the segmented homogeneous components which gives us insight about the performance of each segmentation modules discussed in section II.

In the matrix we can see that other than a few cases, the overall performance is satisfactory. The reason behind image and text segments gaining greater false positive and false negative values is low pixel density of some images and high

	Image	Text	Table	Cell
Image	84	9	0	0
Text	27	220	3	0
Table	1	6	53	3
Cell	2	20	6	224

TABLE I
CONFUSION MATRIX

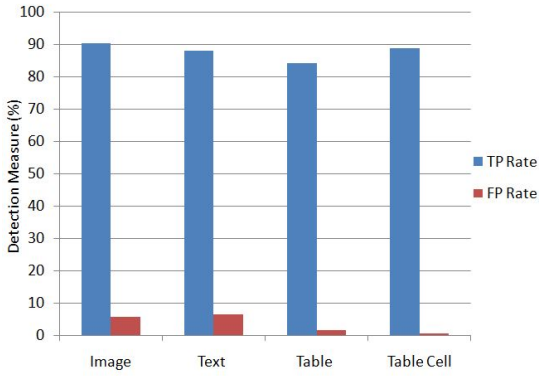


Fig. 7. TP rate and FP rate column chart for homogeneous components.

pixel density of some texts. For this reasons, sometimes images are identified as texts and texts as images. It can also be observed that several table cells have been identified as text segments. In this case a table was not correctly detected and it was left behind in the text region. As a result cells in that table were counted as text components which made the count of false positives of texts that is, the count of false negatives of cells greater.

Calculation of TP Rate and FP Rate: TP rate and FP rate were calculated from the values we obtained in the confusion matrix of Table I. These rates were put in the column chart against the segmentation instances in figure 7.

- In figure 7, it is observed that **True Positive Rate** is above 80% for all the modules. Images have the highest measure of 90%.
- It can be assumed from figure 7 that **False Positive Rate** is not in a destructive state for the modules, but images and texts are detected more falsely compared to the other two components. False detection of texts are more often images which have hollow contours because they somewhat lack halftone properties. Tables have a very low rate of False positive which is about 2%.

We can see from the above analysis that, the TP Rate is much higher and FP Rate is not that much alarming. So it can be assumed that components can be detected and document pages can be segmented accurately by our proposed system.

The data-set we used consists of images originating from different sources namely digitally generated pdf, scanned pdf from online sources, manually scanned data. In figure 8, accuracy for each type of components are shown using a column chart. Table and cell detection have a higher accuracy of 96.8% and 95% respectively whereas image and text detection have 90.70% and 90% accuracy.

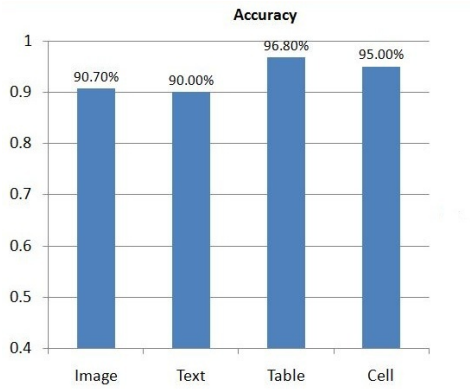


Fig. 8. Percentage of Accuracy in Detecting Components

IV. CONCLUSION

This paper proposed segmentation method which can separate texts, images and tables using morphological operations. It is language independent and also does not require any explicit training phase or any sort of pre-labeled data. Our proposed RIFR (Residual Image Fragments Retrieval) and TETC (Text Extraction from Table Cells) method improved the performance of the segmentation significantly.

Though using morphological operations has its benefit, but its accuracy is not up to the mark for all kinds of documents. Specially, halftone images with hollow contours cannot be separated from texts. Also horizontal and vertical lines which are not part of the table but is connected with it are sometimes identified as part of the table.

Future improvements may include better identification of image and tables. We also aim to work on separating texts irrespective of size and composite components like extracting text from images.

REFERENCES

[1] Sébastien Eskenazi, Petra Gomez-Krämer, and Jean-Marc Ogier. A comprehensive survey of mostly textual document segmentation algorithms since 2008. *Pattern Recognition*, 64:1–14, 2017.

[2] Karin Sobottka, Horst Bunke, and Heino Kronenberg. Identification of text on colored book and journal covers. 06 1999.

[3] Anil K. Jain and Yu Zhong. Page segmentation using texture analysis. *Pattern Recogn.*, 29(5):743–770, May 1996.

[4] Thotreingam Kasar, Philippine Barlas, Sébastien Adam, Clément Chatelain, and Thierry Paquet. Learning to detect tables in scanned document images using line information. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 1185–1189. IEEE, 2013.

[5] Dan S. Bloomberg and Sun Sparcstation. Multiresolution morphological approach to document image analysis. In *1th International Conference on Document Analysis and Recognition (ICDAR 91*, 1991.

[6] Thomas Breuel Syed Saqib Bukhari, Faisal Shafait. Improved document image segmentation algorithm using multiresolution morphology. SPIE, 1 2011.

[7] Vikas J. Dongre and Vijay H. Mankar. Devnagari document segmentation using histogram approach. *CoRR*, abs/1109.1247, 2011.

[8] Youbao Tang, Xiangqian Wu, and Wei Bu. Text line segmentation based on matched filtering and top-down grouping for handwritten documents. In *Document Analysis Systems (DAS), 2014 11th IAPR International Workshop On*, pages 365–369. IEEE, 2014.

[9] Open Source Computer Vision Library. https://en.wikipedia.org/wiki/OpenCV.