Detection and Segmentation of Text in Handwritten Hindi Documents

Thesis submitted in partial fulfillment of the requirements for award of the degree of

Master of Technology

in

Computer Science and Applications

Submitted By
Rohit Mittal
Roll No. 601303025

Under the Supervision of:
Mr. Khushneet Jindal
(Sr. System Analyst)

Computer Science and Engineering Department
Thapar University,
Patiala-147004

June, 2015

Certificate

I hereby certify that the work which is being presented in the thesis entitled, "*Detection and Segmentation of Text in Handwritten Hindi Documents*", in partial fulfillment of the requirements for the award of degree of Master of Technology (M.Tech) in Computer Science and Applications submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of Mr. Khushneet Jindal and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

02/07/2015

Signature

(Rohit Mittal)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

02/07/2015

(Mr. Khushneet Jindal)

Sr. System Analyst

Computer Science and Engineering Department

Countersigned by:

(Dr. Deepak Garg)

Head

Computer Science and Engineering Department

Thapar University

Patiala

(Dr. S.S. Bhatia)

Dean (Academic Affairs)

Thapar University

Patiala

Acknowledgement

First and foremost, I would like to thank my supervisor **Khushneet Jindal**, Sr. System Analyst of Thapar University for giving me an opportunity to work under him on this challenging and burning topic and providing me ample guidance and support through the course of this research. I also thankful to him for inspiring my interests in Optical Character Recognition (OCR). I am grateful for his guidance, encouragement and support throughout my M.Tech research work at Thapar University and also in my life. This thesis comes from his helpful discussions, supervision and meticulous attention to details.

I would like to thank **Dr. S.S. Bhatia**, Professor and Dean of Academic Affairs, Thapar University, Patiala, for giving provisions of the entire required infrastructure such as computer labs, library facilities, immensely useful for learners to equip themselves with the latest in the field.

I take this opportunity to express my appreciations towards **Dr. Deepak Garg** for his kind help and cooperation, and other faculty members for their constructive suggestions through out the research. I am also thankful to all my respected teachers in the department and all my friends, for their direct or indirect help, inspiration and motivation. I am grateful to all the technical and non-technical members of the department for their support.

I want to express my greatest gratitude to my dear mother and father, my loving sisters for their endless love, constant support, encouragement and patients throughout my research without which I would not have reached this position.

Last but not least, I would like to thank almighty for everything.

Rohit Mittal 02/07/2015

Date: 02-July-2015

Place: Thapar University, Patiala

(Rohit Mittal)

Abstract

Optical Character Recognition(OCR) is one of the most important applications of Computer Science. The accuracy of Character Recognition in the case of machine printed characters has improved, but in case of Handwritten documents researchers are still working on accuracy improvement. The system is complex as different writers have different way of writing and even in the same document a single character or word can be represented differently by the same writer, which leads to less accurate results.

In said environment, accurate segmentation plays a very important role. To achieve this, one has to efficiently handle problems like unequal spacing between text lines, overlapped text lines, connected components between the lines, unequal height of characters, separation of upper and lower modifiers. During the segmentation stage, if some error occurs then it will keep on increasing the error rate in the next stages. In this work, new Segmentation algorithm(s) has been proposed for Handwritten Hindi documents. An encouraging segmentation accuracy rate of 96.87% at line, 92.45% at word and 87.13% at character level has been achieved.

Contents

List of Figures

List of Tables

List of Algorithms

Glossary of Abbreviations/Symbols

List of abbreviations used throughout the thesis:

- A
  - ⋆ ANN: Artificial Neural Networks
- D
  - ⋆ DPI: Dots Per Inch
- H
  - ⋆ HCR: Handwritten Character Recognition
  - ⋆ HMM: Hidden Markov Models
- O
  - ⋆ OCR: Optical Character Recognition
- S
  - ⋆ SVM: Support Vector Machines
- T
  - ⋆ TSD: Tightly Spaced Documents
- W
  - ⋆ WSD: Widely Spaced Documents

1

Introduction

1.1 Introduction

Optical Character Recognition (OCR) is the process of converting the scanned images of handwritten or machine printed text (letters, numerals, and symbols), into a computer format (such as ASCII) text [2]. The scanner is use to produce an image of the document, just like taking a picture of it. The OCR software then examines the Scanned image of the document; identifies each number, letter, and punctuation mark; and generates equivalent text in a machine-readable form that can be used by a computer system. So one can say that the OCR system take a scanned image of paper documents as input and produce a symbolic text document (*e.g.* Word, ASCII, or HTML) [3] as output.

A large amount of handwritten data is stored in the form of files in offices or companies. This handwritten data can be digitized manually by a data entry operator which is not a favorable option both in terms of time and money. A better option to convert these data into computer readable format is by using optical character recognition (OCR) so as to make it eligible for editing, searching and addition in a computer system. Also for digital libraries such as Universal Digital library, Digital Library of India [4] OCR plays an important role for digitization of paper documents. Nowadays, the process of digitization from handwritten documents into computer editable form is going on in sectors like government offices, libraries, banks, colleges, *etc.* So the OCR system is developed to convert the scanned documents into computer editable format. Major application of OCR includes (i) Signature verification and identification (ii) Reading bank deposit slips (iii) Automatic number-plate reader (iv) Make electronic images of printed documents search-able [1].

Optical Character Recognition is broadly categories into two sub-fields as shown in Fig. 1.1 (i) Machine Printed Character Recognition (ii) Handwritten Character Recognition.

1.1.1 Machine Printed Character Recognition

Recognition of Printed text includes scanned images of books, newspaper *etc.* as shown in Fig. 1.2.

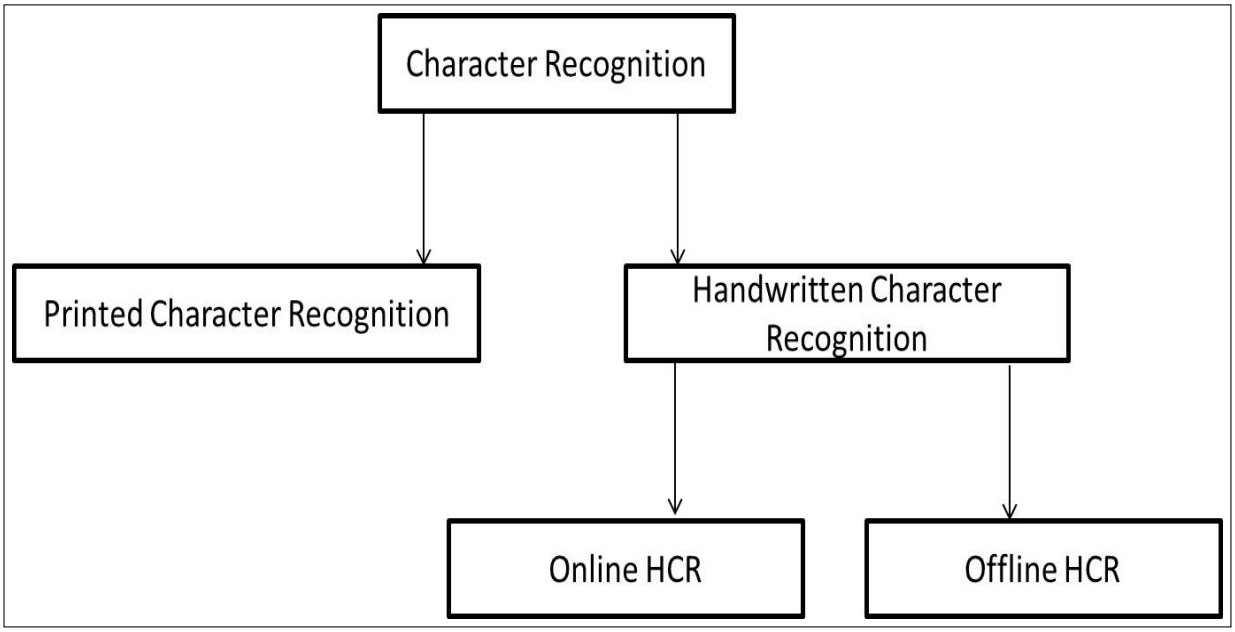


Figure 1.1: Types of OCR

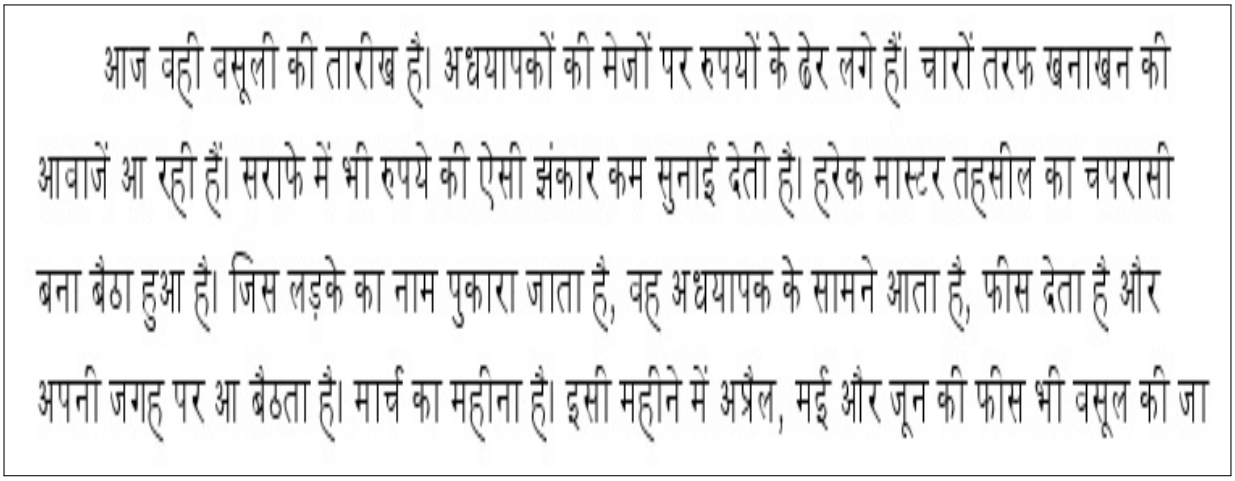

आज वही वसूली की तारीख है। अध्यापकों की मेजों पर रुपयों के ढेर लगे हैं। चारों तरफ खनाखन की आवाजें आ रही हैं। सराफे में भी रुपये की ऐसी झंकार कम सुनाई देती है। हरेक मास्टर तहसील का चपरासी बना बैठा हुआ है। जिस लड़के का नाम पुकारा जाता है, वह अध्यापक के सामने आता है, फीस देता है और अपनी जगह पर आ बैठता है। मार्च का महीना है। इसी महीने में अप्रैल, मई और जून की फीस भी वसूल की जा

Figure 1.2: Hindi machine printed text sample

1.1.2 Handwritten Character Recognition

Handwritten character Recognition is the recognition of characters which are written by writers as shown in Fig. 1.3. Handwritten text can be further divided into two categories. One is offline Handwritten character Recognition in which, the writer writes on paper using pen or pencil. Another type of Handwritten character Recognition is an online Handwritten character Recognition, in which the writer writes using some digital equipment such as electronic pen.

HCR is difficult to recognize due to the very large variation's associated with different handwriting styles of different individuals. Hence, existing recognition techniques alone are not enough to develop successful automated solution for this problem. In the last few

decades, various machine learning techniques have been applied to develop procedures for both offline and online HCR. These methods include Support Vector Machines, Artificial Neural Networks (ANNs), Hidden Markov Models (HMMs), Fuzzy Logic, Gaussian Mixture Models, *etc*. The HCR system should have the following characteristics:

1. Flexibility: The system should handle all the variations in writing associated with a wide range of character patterns of different individuals.
2. Customization: The OCR System should have the ability to learn new user-specific handwritten patterns online, *i.e.,* an individual could be able to customize the OCR system for the one's handwriting style.
3. Efficiency: Online HCR systems should be very efficient in terms of time and space.
4. Automatic Learning: The OCR system should be trained using an automatic learning mechanism in order to provide the customization feature.

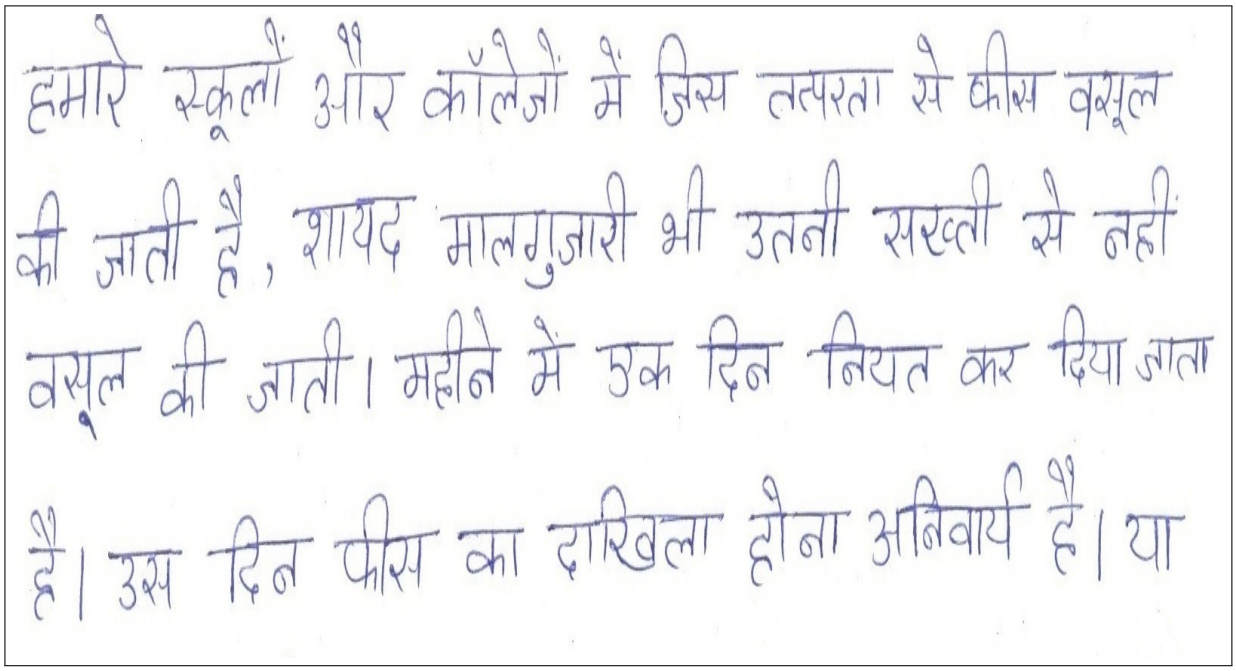


Figure 1.3: Hindi handwritten text sample

1.2 Application of Character Recognition System

The Character Recognition System have a large number of applications that are explained below:

1. **Data identification:** The OCR system can be used for reading the data from some forms *e.g* tax forms, account number verification *etc.*. when the documents are of

same size then it become easy for the OCR system to detect the area where required data is present.

2. **Reading addresses:** We can also read the addresses which are present on the envelop using the ocr system and this address can be used for locating the destination address. This can be used to find out the address on the envelop, read the complete address block, recognize the ZIP code and then generate then bar code using this information.

3. **Reading passport data:** The ocr system can be used for reading the information on the passport like birth date of the person, passport number on the passport *etc.* and this information can be validate against the database records that contains information about criminals.

4. **Automatic number plate Recognition:** The ocr system can be used for the identification of the unique number which is present on the vehicle registration plate.

5. **Bill Processing Readers:** These types of reader are used to read bills like payment slips. The reader try to find out the area where the information like account number and payment value is present.

6. **Legal:** In many industries, there is a need for digitizing the paper documents. In such case, the ocr system simplifies the process by converting these documents into compute editable form.

7. **Health care :** In the medical field ocr system plays an significant role for digitizing paper documents. In this field, they needs to deals with large amount of paper documents like insurance form, patient data entry form *etc.*. Now, ocr system is able to extract these information from forms and convert them into text file that can be easily process by the computer system.

8. **Page Readers:** Page Reader are of two types: High-end page readers(HEPR) and Low-end page readers (LEPR). HEPR are good in the identification of the text. LEPR are fitted with many flat-bed scanners. They are highly used in offices where high throughput is not required because their throughput is less as compare to HEPR. Some popular OCR system grant users to fit the recognition engine with the customer data, which helps to improve the recognition accuracy.

9. **Quick Digital Search:** The ocr system is used to convert any scanned image into the computer editable format. This enables us for finding or searching the text into the files. We can also able to edit the text in the file.

1.3 Introduction of Devanagari Script

After English and Chinese, Hindi is the third most widely used language and all over the world, there are approximately 500 billion people who write and speak Hindi. In India, Devanagari is the basic script of many languages like Sanskrit and Hindi. Hindi language have 11 swar (vowels) as shown in Fig. 1.4, 33 Vyanjan (consonants) [3] as shown in Fig. 1.5 and 12 modifiers (special symbols) as shown in Fig. 1.6. Almost all the words are combination of these two along with the modifiers placed on left, right, bottom and above. It is written from left to right.

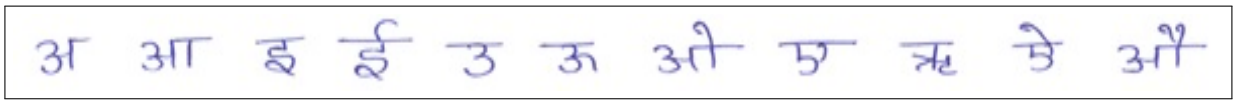


Figure 1.4: Swar in Hindi language

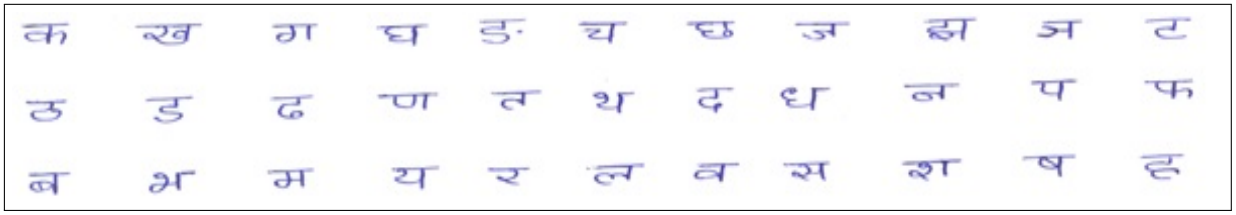


Figure 1.5: Vyanjan in Hindi language

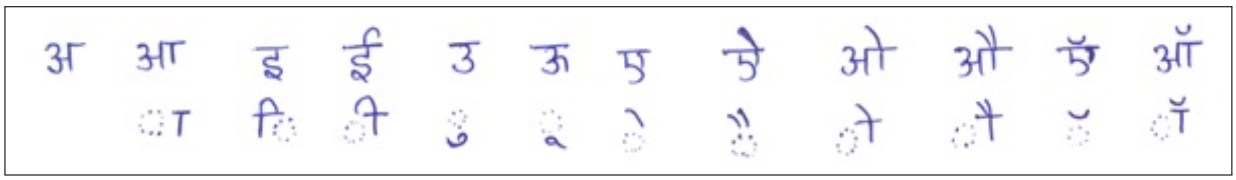


Figure 1.6: Modifiers in Hindi language

Every word is divided into three parts, one part is above the header line includes upper modifiers, second part is core part which includes the characters and the third part is lower part which includes the lower modifiers in the word as shown in Fig. 1.7.

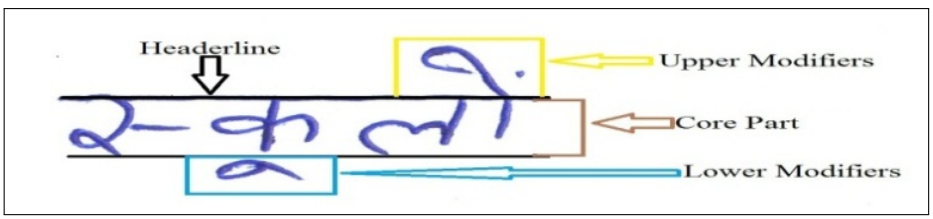


Figure 1.7: Sample word in Hindi language

1.4 Stages in OCR System

The OCR system accepts scanned images as input. The quality of the input image is determined by measuring the number of DPI (dots per inch). Image quality is directly proportional to DPI. We scanned the images at 300 DPI to digitize documents for our experiment because 300 DPI produces sufficient image quality for OCR. If the resolution is too high, then it will take more time for processing and requires more memory space. Recognition of devanagari script includes the following steps as shown in Fig. 1.8.

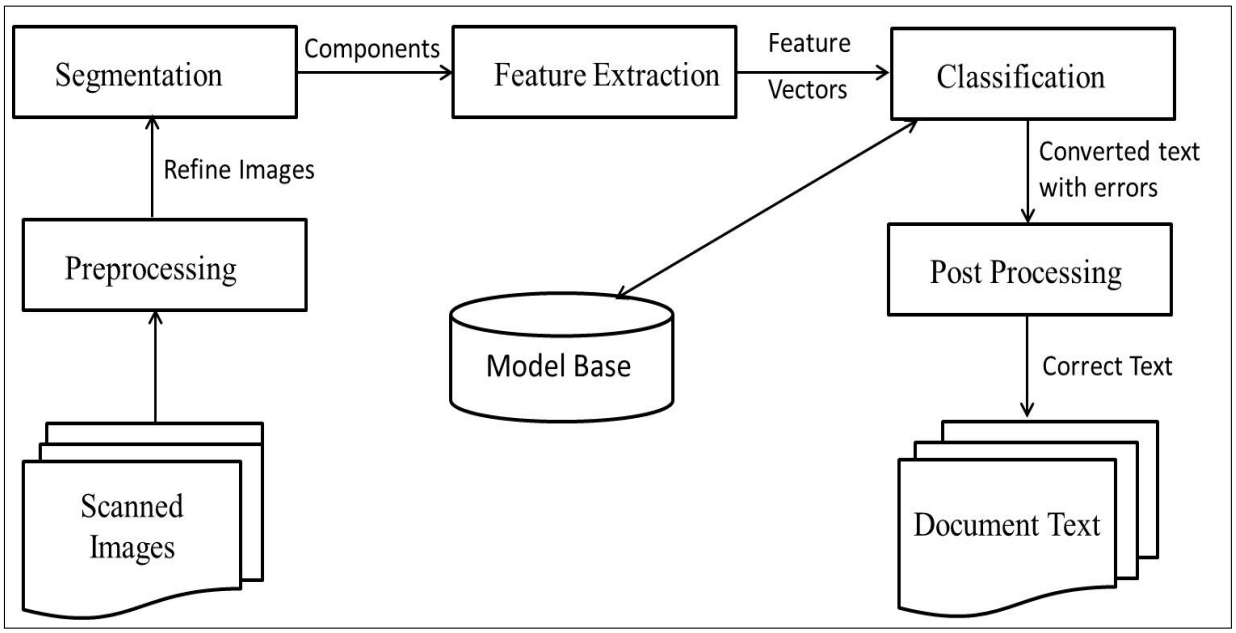


Figure 1.8: Steps in recognition of Hindi Documents [1]

1.4.1 Pre-Processing

This step is used to enhance the document quality. There are a number of actions performed on documents in Pre-Processing phase, *viz*.

Binarization

All scanned images are usually represented as a collection of pixels having intensities which can be represented by integer values from 0 to 255. In the Binarization process, the image as shown in Fig. 1.9 is converted into a binary image as shown in Fig. 1.10 based on some threshold value, if the intensity of a pixel is greater than a particular threshold value, then the pixels are set to white(1), otherwise to black(0). The method that convert the color images into binary form is known as Binarization.

In Binarization, the images are converted into binary form by selecting a threshold value. The threshold value varies according to the quality of the image being used. In our

experiment, the Otsu's method is used, which is implemented using gray-thresh function in Matlab [5] for binarization. Thresholding is categories into two types:

1. Global thresholding: Select a single threshold value for the entire document image.
2. Adaptive (local) thresholding: It uses different values for different set of pixel according to the local area information.

Adaptive thresholding is used in the works that involve images of varying level of intensities, such as scanned medical images. For images like machine printed documents, where the characters are written on a white background, using a global threshold method would be sufficient to distinguish the foreground and the background.

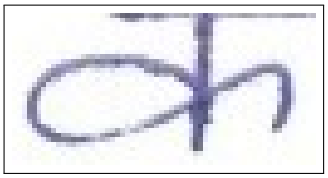


Figure 1.9: Character sample of Hindi text

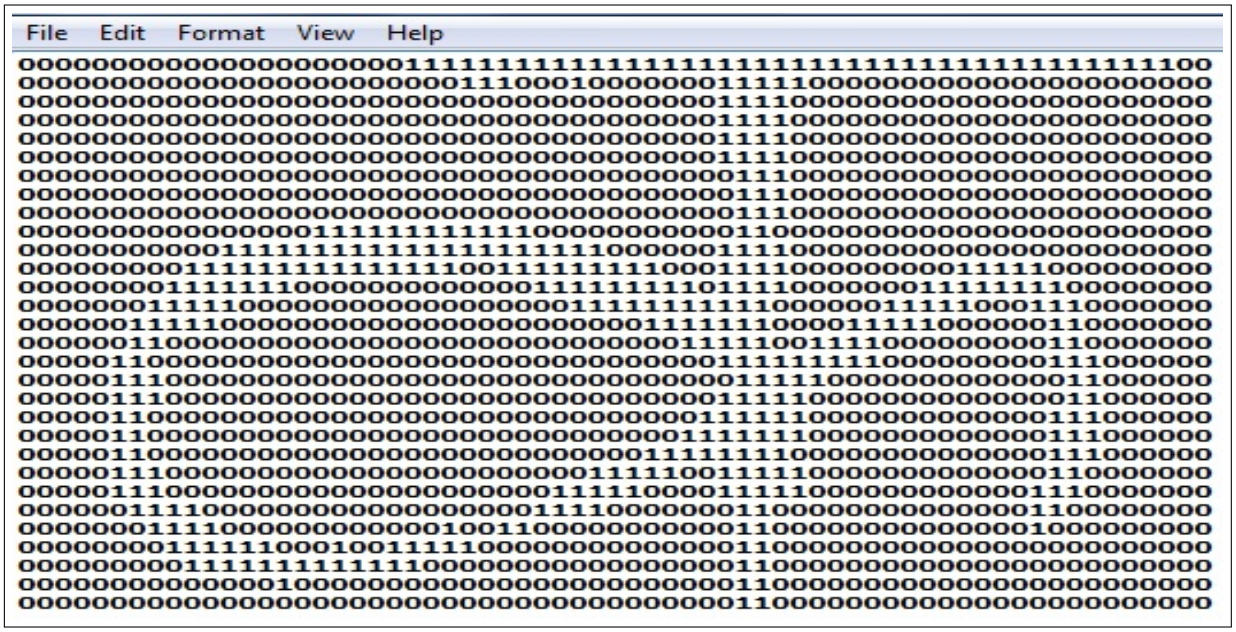


Figure 1.10: Binary representation of character shown in Fig. 1.9

Noise Reduction

Image quality improvement has been a concern throughout in image processing. Noise in an image is unwanted pixels because it degrades image quality. The image resulting from the scanning process may contain a certain amount of noise as shown in Fig. 1.11. The noise, which is introduced by the scanning causes gaps in lines, disconnected line segments

etc. The deformation in images, including rounding of corners, local variations is also the factor that causes noise which degrades the pattern of an image. For successful character recognition, it is necessary to get rid of these noise. There are two types of techniques for removing the noise in the document:

1. Filtering: This aims to remove noise that is introduced by uneven writing surface. Filters are designed for sharpening the image and for removing the colored background in the image.

2. Morphological Operations: In these types of operations, one has to filter the image and replace the convolution operation by the logical operations. There are large number of morphological operations which are designed for the decomposition of the connected strokes, smooth the contours connect and the broken strokes.

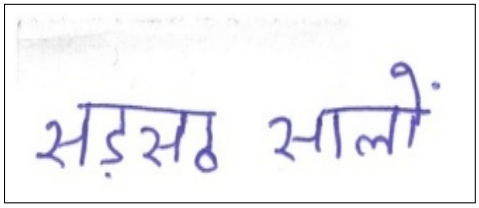


Figure 1.11: Input text image with noise

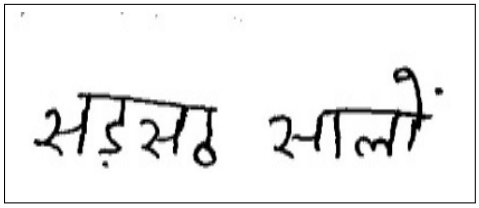


Figure 1.12: Text image after noise removal

Skew Detection and Correction

When we scan a document, if it is not aligned correctly then it results in a skewed image as shown in Fig. 1.13. Skewed angle is the angle that the lines in the image makes with the horizontal direction. The skew correction in an document image still remains a challenging task. The skewness of the document image should be corrected accurately to the level that is accepted without complaints as shown in Fig. 1.14. Hough transforms, Projection Profile Fourier transformation methods are widely used for this purpose.

Slat Normalization

Slant angle in the angle between longest stroke in a word and the vertical direction as shown in Fig. 1.15. Slant normalization is used to remove the slat in all characters to a

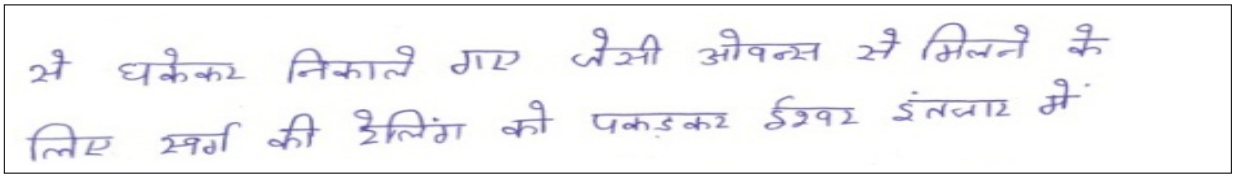


Figure 1.13: Input text image with skew

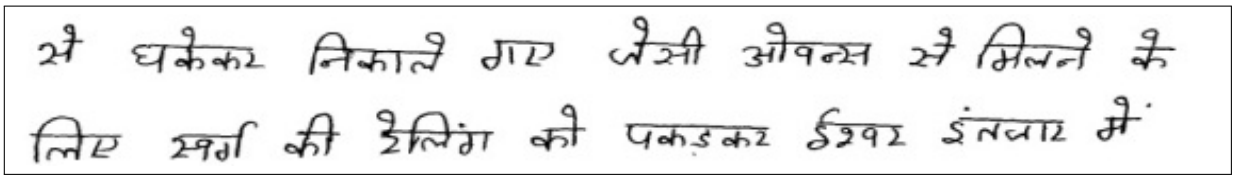


Figure 1.14: Text image after skew removal

standard form as shown in Fig. 1.16. The mostly used procedure for the estimation of the slant is finding the average angle of the nearest vertical character. The coordinates of the slant angle are determined by detecting the start and end points of each line. The Hough transform is used by scanning from left side to the right across the image document.

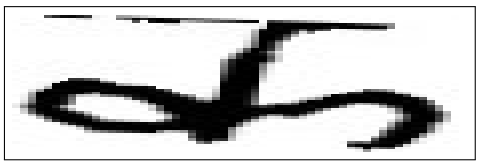


Figure 1.15: Character image with slant

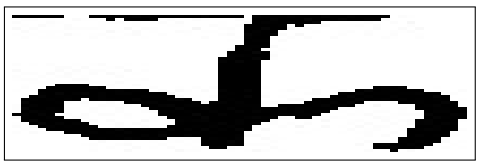


Figure 1.16: Character image after slant removal

1.4.2 Segmentation

The pre-processing phase gives us an clear document in the sense that a high compression, and low noise is obtained in a normalized image. The next stage is segmenting the document. In segmentation phase, information of lines, words and characters is retrieved from the binary image. Segmentation is the building block of any accurate optical character recognition. Segmentation is an important stage because if errors occur during this step, then it will keep on increasing the error rate in its next steps. The main goal of text segmentation is to obtain partitioning of an image into a set of disjoint regions that are different, homogeneous and meaningful with respect to some characteristics. Segmentation is categorized into two types:

1. External Segmentation: The isolation of various writing units, such as sentences, paragraphs, or words is performed under this type of segmentation.

2. Internal Segmentation: The isolation of letters is performed under this type of segmentation.

1.4.3 Feature Extraction

In this phase, one captures the essential characteristics of the symbols and identify the group of parameters which will define the shape of a character uniquely. When the data which is given as an input to the algorithm is too large to process, then this data is transformed into reduces set of features.

The process of transforming such data into set of features is called feature extraction. The features to be extracted are carefully chosen such that the feature set will extract the relevant information from the input data in order to perform the desired task using this reduced representation instead of the full size. We extract features from separate segments and one segment is transformed into one feature vector.

1.4.4 Classfication

In classification process, Initially the identification of the character is performed and after that these characters are assigned to the correct character class for accurate recognition. According to the type of learning procedure which is used to generate the output character recognition is categorized. According to the supervised learning the training set consisting of a set of instances that have been appropriately labeled by hand with the correct output.

1.4.5 Post Processing

Post-Processing is the final step of the OCR system. The main objective of this phase is to detect and correct the grammatical mistakes in the OCR output. There are two types of OCR errors that are corrected in this phase: real word problems and non word problems.

A real word problem is a word that is identified by the OCR and have an entry in the dictionary; the recognized word is grammatically incorrect *w.r.t.* the line in which the word is present. For example, if "How are you" is identified by the OCR system as "How are your", then "your" is the real word problem because "your" in the sentence is grammatically incorrect. A non word problem is a word that is identified by the OCR; but there is no entry of the respective word in the dictionary. For example, if "How are you" is identified by the OCR system as "Huw are you", then "Huw" is a non word problem because there is no entry in the dictionary for the respective word . These problems are categories into three classes:[6]

1. Deletion Problems

2. Insertion Problems

3. Substitution Problems

1.5 Outline of Thesis

The primary objective of this dissertation is to be evolved an offline handwritten character recognition system for Hindi Language. The present study represents the ways in which the system can recognize the Offline handwritten characters written by different writers. In this chapter, we have given the brief introduction of the following:

1. Optical character Recognition System (OCR) and the various steps which involve in the OCR.

2. I have done my work on Devanagari script so we also give the brief introduction of Devanagari script.

In chapter 2, we give a brief introduction about the research papers which I studied during my research. This provides an essential knowledge of offline handwritten OCR. In chapter 3, we have discussed various problems which we faced during our research with their possible solutions. In chapter 4, we have discuss proposed algorithm(s) which handle various steps involves in segmentation *i.e.*, Line Segmentation, Word Segmentation and Character segmentation. In chapter 5, the results achieved by these algorithms are compared with the existing techniques. In chapter 6, we present conclusion and the future work.

2

Literature Review

During recent years, research is getting increasing towards handwritten devanagari character recognition, although the first research paper on handwritten devanagari character recognition was published in 1977 [7], many researchers recently proposed various techniques. For example, **N. Garg** et al. [8] proposed a method for the segmentation of lines in handwritten Hindi text. In this method firstly they detect the header line in the word followed by a contour technique in which, he divide the overlapped lines.

In [9], **B. Chaudhuri** et al. proposed a technique which is based on the dependencies between inter line and text line gap. At the first step, they try to find out the text and inter line gap points and then draw the curve using these points. The curve starts moving from one side of the document and continuously move to the other side while a point is used to guides the curve, so that the curves never touches to each other.

In [10], **R. Kumar** et al. proposed a technique to segment Gurmukhi handwritten text. In this technique, the whole image is assumed as a large window, then this window is broken into smaller size windows which gives lines. On successful identification of lines, each window is used to segment the words present in those lines and finally characters are segmented from those words.

Y. Chen et al. [11] proposed a procedure for segmenting single or multiple touching handwritten numeral string. Most of the algorithms for the segmentation of connected components focus on the analysis of foreground pixels. In this paper, they combined background and foreground pixels for the segmentation of single or multiple touching handwritten strings. Many probable segmentation procedures are then constructed and all the ineffective strokes are removed. It the last step, the parameters of geometric properties of each probable segmentation procedure are determined and these parameters are analyzed by the mixture Gaussian probability function for deciding the best segmentation procedure.

N. Arica et al. [12] had studied a lot and progressed to a level sufficient to produce a technology driven applications of Optical Character Recognition. Nowadays, the faster growing computational power enables the implementation of the present CR procedure and creates an increasing demand on many emerging application domains, which require

more superior methodologies.

P. Slavik et al. [13] provides different procedures for slant and skew corrections. In their work, they shows that, if we correct the skew by rotation followed by correction of slant with the help of shear transformation in the horizontal direction is equivalent the correction of slant by a shear transformation in the horizontal direction followed by correction of the skew by shear transformation in the vertical direction.

U. Garain et al. [14] presented a new approach for identification and segmentation of touching handwritten characters. Error in character segmentation is mainly due to the poor recognition OCR system. To design an effective character segmentation procedure, existence of touching characters in the scanned documents has to be handled without errors. The technique provided in this paper is based on fuzzy multi factorial analysis. To segmenting the touching characters in scanned documents, a predictive algorithm is developed which effectively select the possible columns for the separation.

Z. Shi et al. [15] proposed a new algorithm for detection and segmentation of text in handwritten documents. The proposed algorithm is based on the application of a fuzzy directional run length. The proposed technique was tested on a variety of handwritten documents scanned images including historical handwritten documents such as Newton's and Galileo's manuscripts and postal parcel images. The accuracy of 93% is achieved on the test set.

R. Manmatha et al. [16] proposed an algorithm for separation of words in handwritten documents. A huge collections of handwritten historical documents are present in many libraries, museums, and other organizations. In the very first step, the page is cleaned to remove margins, then for finding lines gray-level projection profile algorithm is applied. An anisotropic laplacian at several scales is then used to filtered all lines in the images. The proposed procedure produces blobs which correspond to a portions of the characters at very small scales and proceeds towards larger scales. Finding the best favorable scale at which blobs correspond to words is a critical stage in the algorithm. This is done by determining the maximum over scale of the extent or area of the blobs. Three different approaches are projected for the scale maximization. To recover the words most favorable scale is bounded by a rectangular box. To reduce boxes of unequal size which are not correspond to words, a post-processing filtering step is performed.

M. Jindal et al. [17] proposed a procedure for the segmentation of horizontally overlapped lines and successfully handles the errors which are generally found in printed Indian scripts. According to there algorithm, complete input image is separated into smaller components called as strips and then the algorithm is applied for the segmentation of horizontally overlapped text lines.

A. Zahour et al. [18] proposed a new technique for the segmentation of text lines on Block Covering. This technique resolves the problem of multi-touching and overlapping components. The Block Covering technique is processed in three steps: a new fractal anal-

ysis (Block Counting) for document classification, a statistical analysis of block heights for block classification and a neighboring analysis for building text lines. The first step *i.e* fractal analysis, associated with a fuzzy C-means scheme, is performed on document images in order to classify them according to their complexity: tightly (closely) spaced documents (TSD) or widely spaced documents (WSD). An optimal Block Covering is applied on tightly (closely) spaced documents (TSD) which include multi touching and overlapping lines. The system successfully segment text lines in the image, even if the text lines are overlapped and touch each other.

G. Louloudis et al. [19] proposed a method for detecting text line in the handwritten documents. The proposed technique is based of three distinct steps. In the first step, image is pre-processed and connected component are extraction, then division of the connected component domain into three spatial sub-domains is performed and average character height is calculated. In the second step, a block-based Hough transform is applied for the detection of potential text lines while in third step author correct feasible splitting, to detect possible text lines that the previous step did not expose. In third step, the vertically connected characters are disconnected and assigned to text lines.

N. Stamatopoulos et al. [20] provides a method which is the combination of different segmentation techniques. To generate improved segmentation results the author exploit the segmentation output of complementary techniques.

Y. Li et al. [21] proposed an algorithm in which the density of the pixels are estimated which is called as level set method. The input document is scanned for finding the probability map where each element represents the probability of the underlying pixel belonging to a text line. After that level set method is applied to determine the boundary of neighborhood text lines by evolving an initial estimate.

A. Nicolaou et al. [22] proposed an approach for the segmentation of handwritten document into the respective text lines by shredding the surface of those lines with local minima tracer. In this approach, author assumed that there exists a path from one side of the image to other that traverses only one text line. In the first step, image is blurred and after that they use tracers to follow the black most and white most gaps from one end of the documet to the other end in order to divide the image into text line.

A. Graves et al. [23] proposed an approach which is based on a novel type of recurrent neural network, specially designed for sequence labeling tasks where the data is hard to segment. Experiments is performed on two large unconstrained handwriting databases and achieved word recognition accuracy of 79.7 percent on online data and 74.1 percent on offline data.

A. Pezeshk et al. [24] proposed an approach which was based upon automatic feature extraction and text recognition from scanned topographic maps. The OCR system has the capability to achieve a recognition rate of 94% for the extracted text.

Yi. Pan et al. [25] proposed an hybrid approach for the detection of robustly and

localize texts in natural scene images. For content-based image, analysis text detection and localization in natural scene images is important. This task is challenging due to the presence of complex background, the non-uniform illumination, the variations of font size and line orientation.

A. Gulhane et al. [26] proposed an algorithm for removing the noise in an document which is present in the document due to environmental changes. The author focuses their work on the noise issues that changes image pixel value either on or off. The noisy pixels are easily identified in grayscale image, but difficult to recognize in RGB image because any color combination with white (pixel on) or black (pixel off) generate other color. They focused on such technique that reduces noise in both gray scale and RGB images with recovery of originality of source image.

Y. Bassil et al. [6] works on post processing phase in the ocr system and proposed the procedure for word correction in the final output of the ocr system. This procedure uses the "did you mean" feature of Google which is a spelling suggester and is available online. The procedure first divides the image into text lines then from these lines words are segmented. These segmented words are sent to the google's search engine. if the result of the engine shows the misspelled word, then they will suggest the possible word which will replace the original incorrect word.

D. Salvi et al. [27] proposed a procedure in which to detect the possible text line, the average of resulting characters is used. To detect the possible locations for the segmentation of neighboring characters, a graph model is used, then they applied the longest path procedure for the identification of the optimal segmentation.

G. Pirlo et al. [28] works on zoning-based classification in which he present a new class of membership functions called as Fuzzy-Membership Functions (FMFs). In this type of classification a membership function defines the way in which feature influences the different zones of the zoning method. In order to maximize the performance of the classification FMFs can be easily adapted to the specific characteristics of a classification problem. In his work, a real-coded genetic algorithm is proposed to find the optimal FMF in a single optimization procedure.

S. Shelke et al. [29] presents a method for the recognition of unconstrained characters of handwritten Devanagari documents. This method is based on the multi stage classification scheme. In classification stages the characters are categorized into smaller groups. In his work, the classification is performed using two stages, one is based on fuzzy inference system and the other stage is based on structural parameters. The accuracy of recognition achieved by the proposed method is 96.95%.

M. Kumar et al. [30] proposed a novel hierarchical technique for the recognition of isolated handwritten Gurmukhi characters. In their work, a feature set of 105 feature elements is proposed and four types of topological features are used, *i.e*, vertically peak extent features, horizontally peak extent features, centroid features and diagonal features.

Support Vector Machines (SVMs) classifier is used in his work for the purpose of classification. The recognition rate of 91.80% have been achieved with proposed technique.

S. Singh et al. [31] developed a method for the recognition of the text. This method is based on the template matching technique using correlation coefficient. For the testing purpose, they had collected a data set of 61 words, which includes almost all the combination of the words that may be written on the bank cheque. The proposed algorithm gives the overall accuracy rate of 76.4% .

A. Gaur et al. [32] works on the recognition of the Hindi characters. The whole recognition is performed in three steps. In the very first step, the image is pre-processed, in order to remove the anomalies in the image like skew, slant, then the image in converted into binary form followed by the separation of characters. The Horizontal bar in Hindi character is also removed in this step. In the second step, feature extraction is performed in which region based k-means clustering is used. In the last step, classification is performed on the data using the support vector machine.

It has been observed from the literature survey that most of the work on OCR has been done for English, Japanese, Chinese, Arabic, Urdu, Devanagari script and Tamil. The recognition systems for Devanagari script are still in initial stages and the most of the work for Devanagari script is carried out on isolated character recognition. This thesis is an attempt to develop an offline handwriting segmentation system for Hindi language, so that recognition of character can be done with more accuracy.

3

Problem Statement

In OCR, Segmentation is an important phase and accuracy of character recognition highly depends on precision of segmentation. In this phase, the segmentation of text line, word, and character is performed. Their are number of challenges in the segmentation phase of OCR, which are classified into three categories, *i.e.*, challenges during line, word and character segmentation.

3.1 Line Segmentation

There are number of challenges associated with the line segmentation as shown in Fig. 3.1:

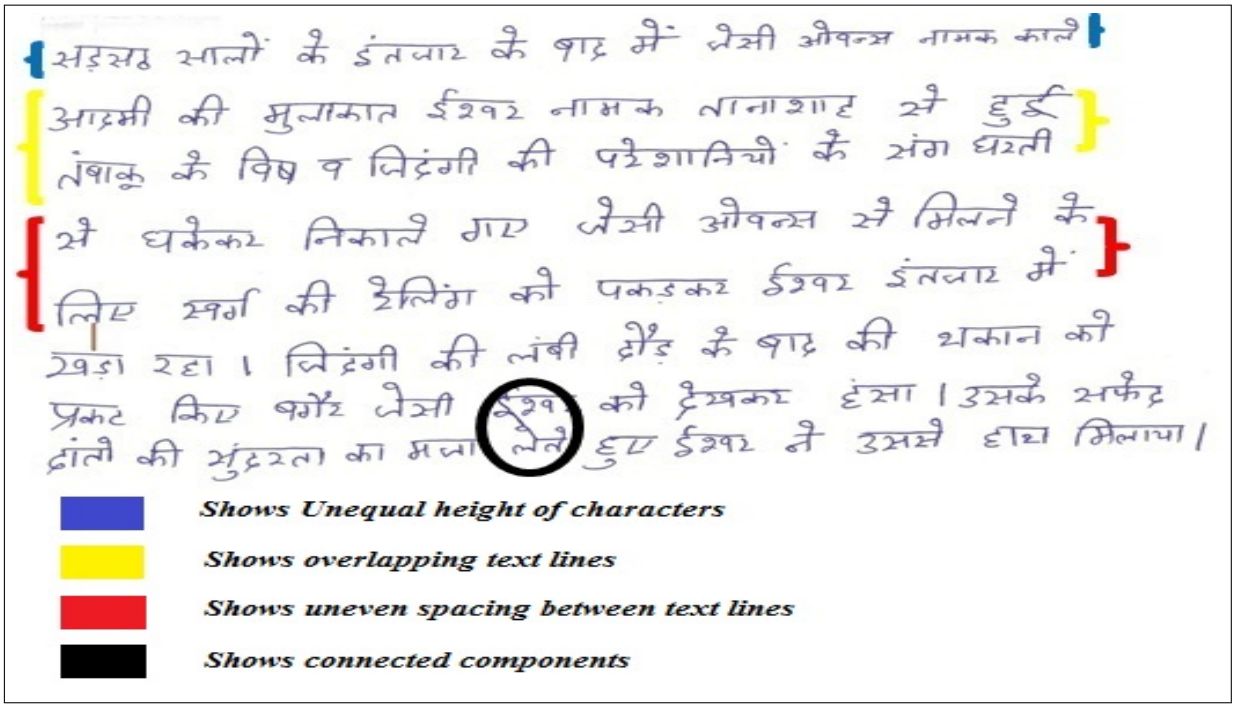


Figure 3.1: Problems identified during line segmentation phase

3.1.1 Unequal Line Height

In machine printed documents, almost all the words are of the same height but in case of handwritten documents, the height of the words varies a lot as shown in Fig. 3.1. As a result, the line may not remain straight and hence leads to an incorrect segmentation.

3.1.2 Text Lines are Overlapped

This is one of the major problems and frequently occurs in cases of roughly handwritten documents. There is a high chance of loss of important information during the separation of overlapped lines as shown in Fig. 3.1.

3.1.3 Unequal Spacing between Text Lines

These are the consecutive lines which are having a vertical varying gap in between as shown using red color in Fig. 3.1. These line appears as they will meet each other.

3.1.4 Connected Components between Lines

Connected component mostly results in loss of important pixel data. It is very difficult to separate connected components as algorithm cannot take suo motu decision on division of connected components *i.e.* how much portion of the connected components goes to upper or lower lines.

3.2 Word Segmentation

3.2.1 Pixels Missing in the Headerline of a Word

In this case, there is loss of pixels in the word due to which, a single word may be taken as two different words as shown in the Fig. 3.2. To handle these types of problems, one can

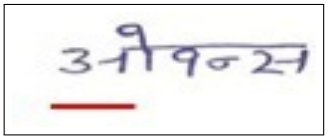


Figure 3.2: Sample of word with missing black pixels in header line

check for the continuous 2-3 vertical white pixel gap during the segmentation of the words.

3.2.2 Connected Words

In this case, Words are connected with each other but appears as a single word as shown in the Fig. 3.3. It is difficult to detect whether two words are connected with each other or not but if there exist a small vertical white pixel gap between two words then it can be easily handle by finding the vertical end to end white pixel gap.

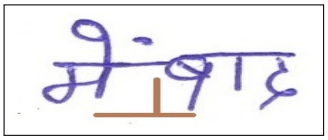


Figure 3.3: Connected words

3.3 Character Segmentation

3.3.1 Compound Characters Problem

In this case, there is a problem of separation of compound characters (*i.e.*, two characters joined together) due to absence of proper gap between the characters as shown in Fig. 3.4. In Hindi documents, there are only few number of compound character cases are present.

Figure 3.4: Sample text showing compound character

To handle these this type of problem without losing the structure of the characters, one can generate proper classes for compound characters during the feature extraction phase.

3.3.2 Compound Character having Space between them

Compound Characters are those in which half characters are glued with full characters. In Fig. 3.5, the characters are not actually connected to each other but a character is present in the in-between space of some other character that will create the problem *i.e.*, how to distinguish these two characters with proper indexing. Now, to handle these type of

Figure 3.5: Sample text showing compound character having in between space

problem one can find out the vertical end-to-end white space gap to detect the possibility of the character and also check for the closed loop area in the upper modifier above the headerline.

3.3.3 Lower Modifiers Covering Neighboring Characters

In this case, it is difficult to detect the gap between the characters due to overlapping of the lower modifiers as shown in Fig. 3.6. To identify these type of cases, find out the average width of the character in the same word and then compare this width with the every possible

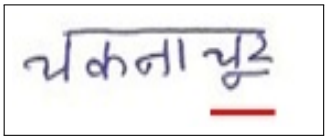


Figure 3.6: Lower modifiers covering neighboring characters

characters in the word, if the width of the character is greater than the average width(or almost double) then the problem exist. To handle this problem, find out the maximum depth of white pixel in the middle area of connected characters. At the bottom of that area proceed to the horizontal direction where we find the maximum white pixel gap, then segment those characters according to the white pixel gap.

4

Proposed Work

4.1 Segmentation

Segmentation is building block of any successful and error free character recognition. In segmentation process the whole image which is in binary form is divided into small meaningful units. In the very first step of segmentation, the document image is divided into individual lines this process is known as line segmentation, further from each of these lines, we try to segment the individual words this process is known as word segmentation and finally, characters are extracted from these words this process is known as character segmentation. The various segmentation steps are shown in Fig. 4.1.

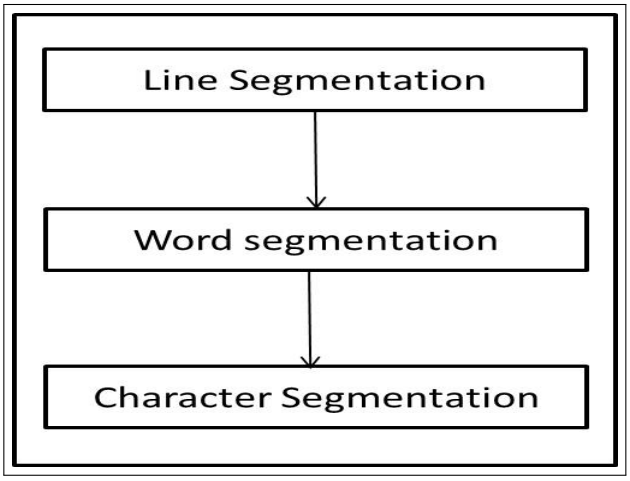


Figure 4.1: Steps involved in segmentation phase

In a text recognition system, character segmentation plays a very important role. In case of printed documents the character may be segmented by determining of the inter character gap but this technique fails in the case of handwritten documents as there are very vast amount of cases are present like Touching characters, Unequal spacing between the characters, Overlapping characters. We had proposed a character segmentation algorithm which successfully handles the over segmentation problems. The various proposed segmen-

tation algorithms, *i.e.* line segmentation, word segmentation and character segmentation are discussed below:

4.1.1 Line Segmentation

The proposed algorithm detects the text lines in the document and successfully handle almost all the problems of line segmentation with high accuracy. The various arrays that are used in the algorithm is shown in Table 4.1.

Table 4.1: List of arrays used in algorithm

Array Name	**Description**
IMG_{Page}	This array consist of colored image which we have to segment
BW	This array contains pre-processed image data which is in binary form
IMG_{Line}	This array contains lines which are segmented using line segmentation alogorithm
IMG_{Word}	This array contains words which are segmented using word segmentation alogorithm
IMG_{Char}	This array contains final output *i.e* segmented characters of the image
$C1, C2, C3$	These are the temporary arrays used in between the algorithm for storing temporary data

To check the efficiency of the proposed algorithm around 120 handwritten samples are collected from different writers of different age group comprising of school and college going students. After data collection all the documents are scanned at 300 DPI gray scale images and proceed those images using the following steps.

Step 1: Initially, the image is loaded into an image array named as IMG_{Page}, then image height and width are read into the variables $Page_{ht}$ and $Page_{wd}$.

$$IMG_{Page} = imread('handwritten.jpg'); \tag{4.1.1}$$

$$[Page_{ht}, Page_{wd}] = size(IMG_{Page}); \tag{4.1.2}$$

Step 2: Image is preprocessed in order to remove the anomalies like noise, skew detection and correction.

Step 3: The image is converted into binary form using Matlab graythresh function [5] and stored that binary data into a new array BD.

Step 4: To detect the line, divide the image into three equal columns arrays C1, C2 and C3, then each column has been searched for clear end-to-end white space gap to detect the lines in the particular column as shown in Fig. 4.2.

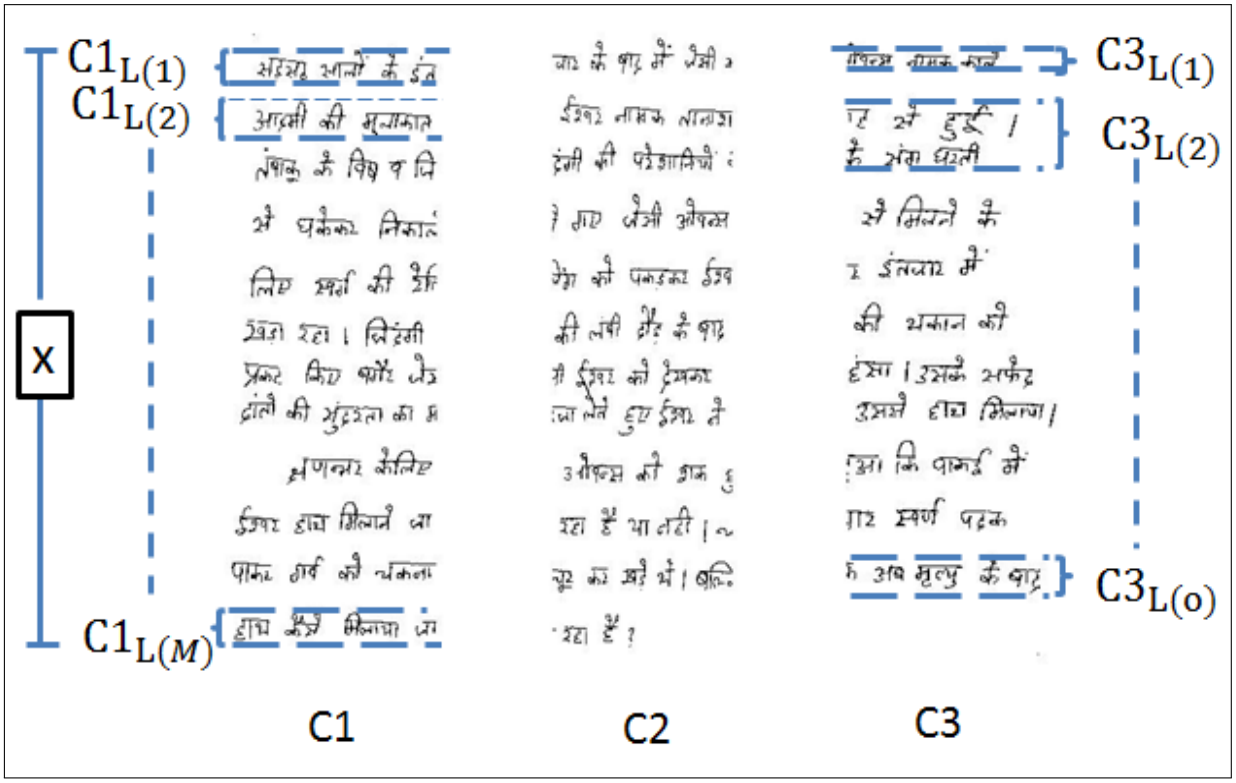


Figure 4.2: Image divided into three columns for finding average height

Now, find the average line height of these three columns independently using the Eq. 4.1.3. Further for more accuracy segmentation result, the average of C1, C1 and C3 has been calculated using Eq. 4.1.6.

$$C1 = \frac{\sum_{i=1}^{m}(C1_{L(i)})}{M}; \tag{4.1.3}$$

$$C2 = \frac{\sum_{i=1}^{n}(C2_{L(j)})}{M}; \tag{4.1.4}$$

$$C3 = \frac{\sum_{i=1}^{o}(C3_{L(k)})}{M}; \tag{4.1.5}$$

$$Avg_{line\_height} = \frac{C1 + C2 + C3}{3}; \tag{4.1.6}$$

where m,n,o = number of possible line in column 1,2,3 respectively,
and m,n,o != 0,
and $C(r)_{l(j)}$=j^{th} line height in r^{th} column.

Step 5: After obtaining the Avg_{line_height}, the variables of BD array are set to the start location, to detect and segment text lines on the basis of average line height using Algorithm 1.

Algorithm 1: Line Segmentation using Average Height Approach

Step 1: Initialize the row variable with 1.

Step 2: Detect the possible line by finding the starting and ending row having end-to end black pixel count is zero.

Step 3: If the line height of the detected line is less than the average line height then go to step 9 else go to step 4.

Step 4: Make a temporary line cut somewhere near to the middle of the detected line. The middle location is determined by adding the start index of the line to the average line height.

Step 5: Find the 3% area above and below to that temporary line cut (as shown in Fig. 4.6).

Step 6: In that area, find out the row having minimum number of black pixels

Step 7: If the minimum black pixels in that row are less than 5% than make the temporary line cut as permanent line cut and go to step 8 else go to step 9.

Step 8: Pass the line to word segment function from start index of the line to the permanent line cut and increment the row variable with the line width.

Step 9: Pass the line to Word segment function for further processing and increment the row variable with the line width.

Step 10: If end of the page *i.e.* $(row >= pageheight)$ then stop else go to step 2 (for next possible line).

Demonstration of the proposed algorithm and some of the line segmentation problems handled by the proposed algorithm are discussed in following cases:

Case 1: $Line\ height <= Average\ line\ height$

When the line height of a probable line is less than or equal to Average line height as shown in Fig. 4.3, then that line is stored in the templine array and passed it to the word function for further processing.

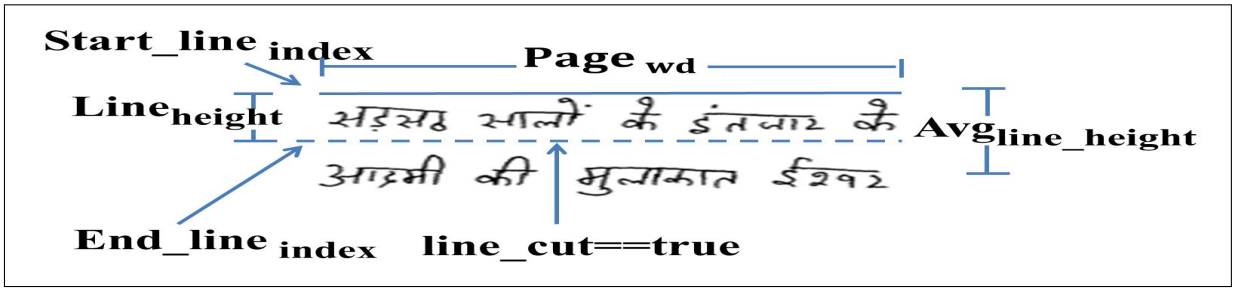


Figure 4.3: Line having line height less than average line height

Case 2: $Line\ height > Average\ line\ height$

When line height of a probable line is more than Average line height then two cases arise, *viz.*

2.1 *Two lines are connected* : When two lines are actually connected as shown in Fig. 4.4.

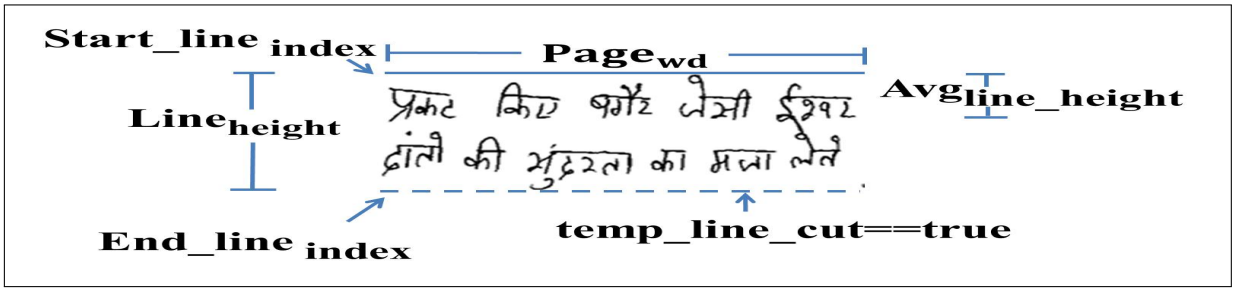


Figure 4.4: Connected lines

2.2 *Single line have more line height than Average height* : Lines are not connected to each other as shown in Fig. 4.5 but due to variations in handwritten documents single line may have more line height than average line height.

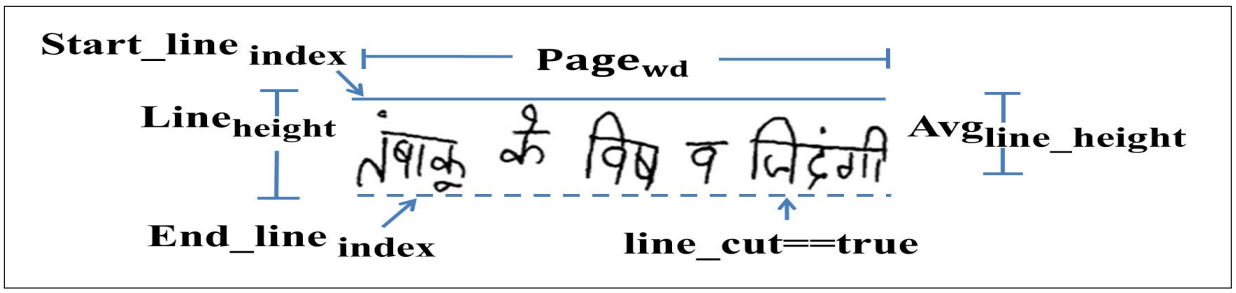


Figure 4.5: Line cut decision in connected lines

In order to differentiate the above cases, following procedure is used:

1. Find the middle location M_{loc} of the image by adding average line height to the starting value of line index.

2. Search 3% above and below area from M_{loc} as shown in Fig. 4.6. In that area, search for row with minimum number of black pixels.

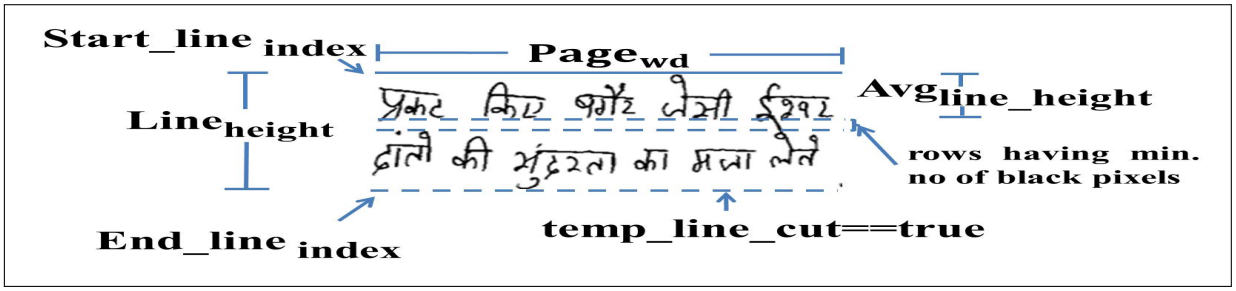


Figure 4.6: Line having line height more than average line height

3. If black pixel count in the selected row is less than 5% than it will be considered as case 2.1.

4. To handle case 2.1, line cut decision is taken to the row having black pixels less than 5% as shown in Fig. 4.7 and that line is stored into temp array and forward to word function for further processing.

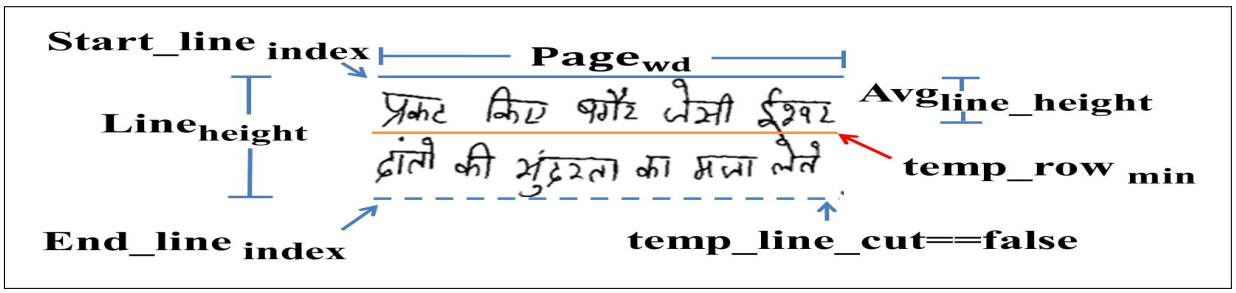


Figure 4.7: Separation of connected lines

5. If the black pixel count in the selected row is greater than 5% then, it will considered as case 2.2, it means that the line is single line written in large size. The sample input image to the line segmentation function is shown in Fig. 4.8 and the output is shown in Fig. 4.9.

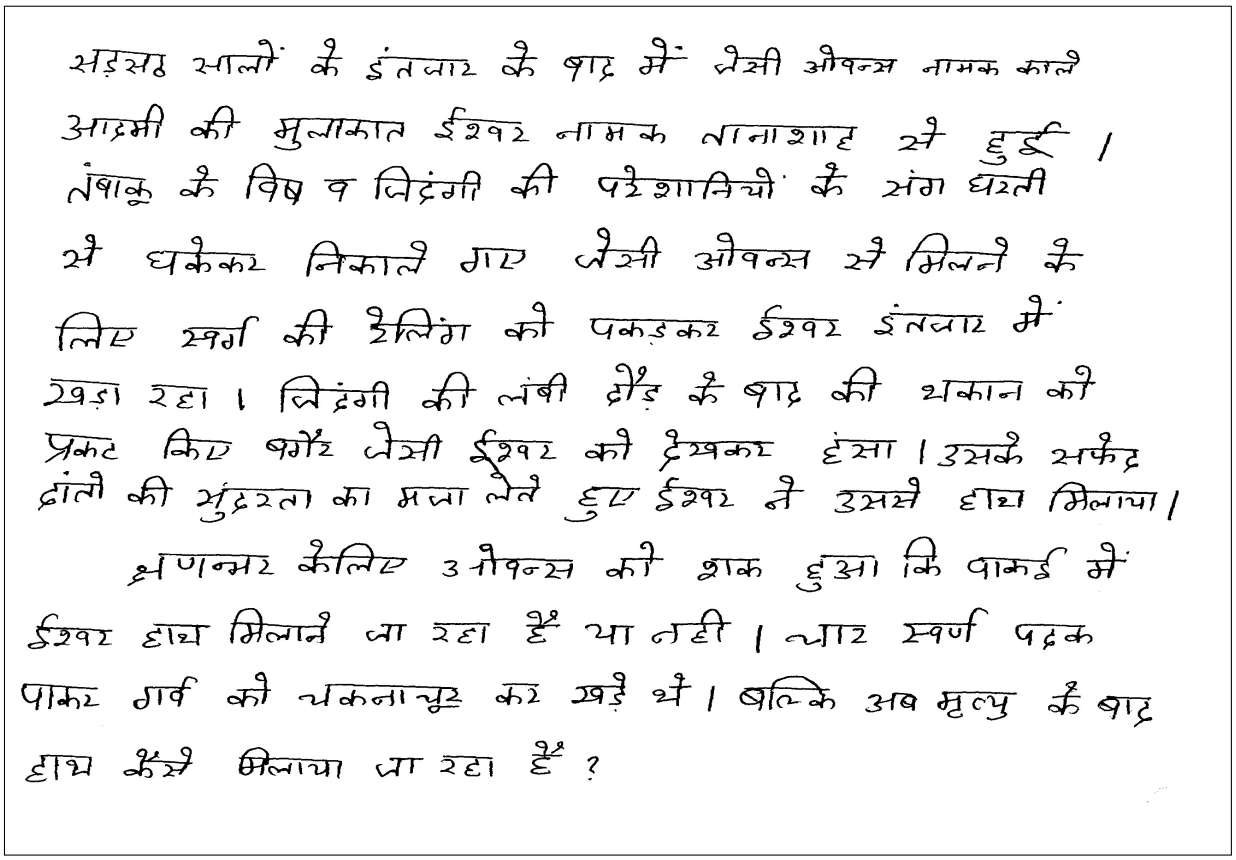
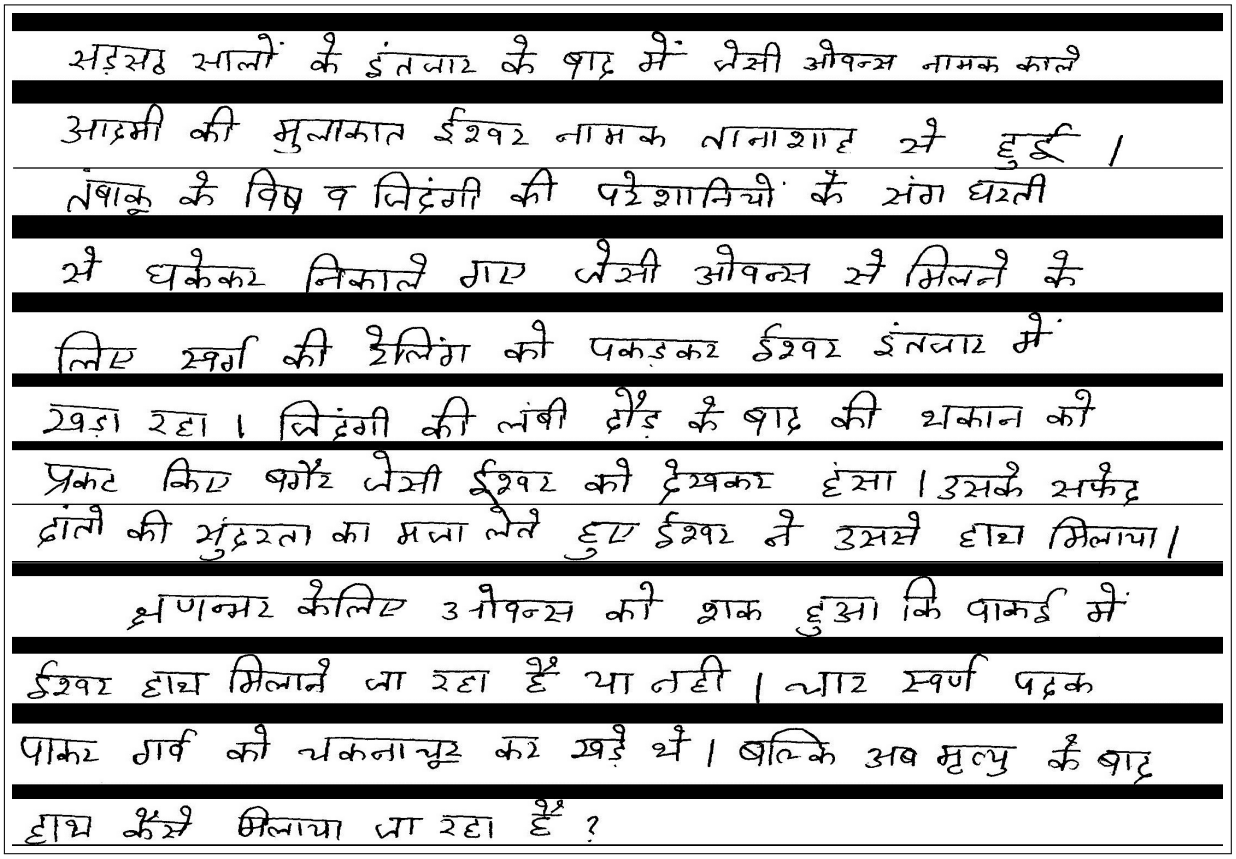


Figure 4.8: Input image to line segmentation function



सड़सठ सालों के इंतजार के बाद में जैसी ओबन्स नामक काले
आदमी की मुलाकात ईश्वर नामक नानाशाह से हुई।
तंबाकू के विष व जिंदगी की परेशानियों के संग घटती
से धकेकर निकाले गए जैसी ओबन्स से मिलने के
लिए स्वर्ग की रेलिंग को पकड़कर ईश्वर इंतजार में
खड़ा रहा। जिंदगी की लंबी दौड़ के बाद की थकान को
प्रकट किए बगैर जैसी ईश्वर को देखकर हंसा। उसके सफेद
दांतों की सुंदरता का मजा लेते हुए ईश्वर ने उससे हाथ मिलाया।
क्षणभर के लिए ओबन्स को शक हुआ कि पासई में
ईश्वर हाथ मिलाने जा रहा है या नहीं। आज स्वर्ण पदक
पाकर गर्व को चकनाचूर कर खड़े थे। बल्कि अब मृत्यु के बाद
हाथ कैसे मिलाया जा रहा है?

Figure 4.9: Output image from line segmentation function

4.1.2 Word Segmentation

It receives the input from line segmentation function in the form of $2-D$ binary array IMG_{Line}. The image height and width are read into the variable $Line_{ht}$ and $Line_{wd}$. In this module, words are segmented from the line received from line segmentation function using Algorithm 2.

$$[Line_{ht}, Line_{wd}] = size(IMG_{Line}); \tag{4.1.7}$$

The input image to the word segmentation function is shown in Fig. 4.11 and the output

Algorithm 2: Word Segmentation Algorithm

Step 1: Initialize count variable to 1.

Step 2: Detect the word by finding the starting column having end-to-end black pixel count is zero and ending columns by finding three continuously columns having end-to-end black pixel count is zero.

Step 3: Store the Word in an array and pass it to character segment function for further processing.

Step 4: Increment the count variable with the width of the detected word.

Step 5: Repeat these steps until count is less than Line Width (*i.e.* end of line)

image from the function is shown in Fig. 4.12. The procedure of word segmentation is discussed below:

1. Starting from the first column to the end of the line, scans the whole column, if vertical end-to-end black pixels in the column are zero, than it will keep increasing the column variable.

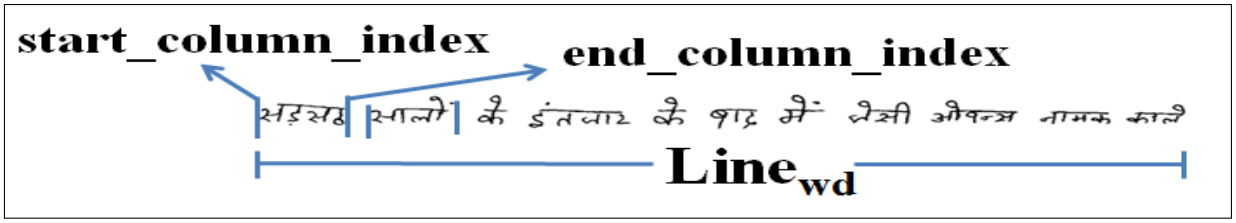


Figure 4.10: Word segmentation example

2. If vertical end-to-end black pixels is not equals to zero, then assign the value of count variable to the start column index.

3. Increment the count variable until we find the three consecutive columns where end-to-end blacks pixels count is not equal to zero, assign its column value to the end column index variable as shown in Fig. 4.10.

4. Extract the word from start column index to the end column index, store the word into an array.

5. Call the character segmentation function for further processing.

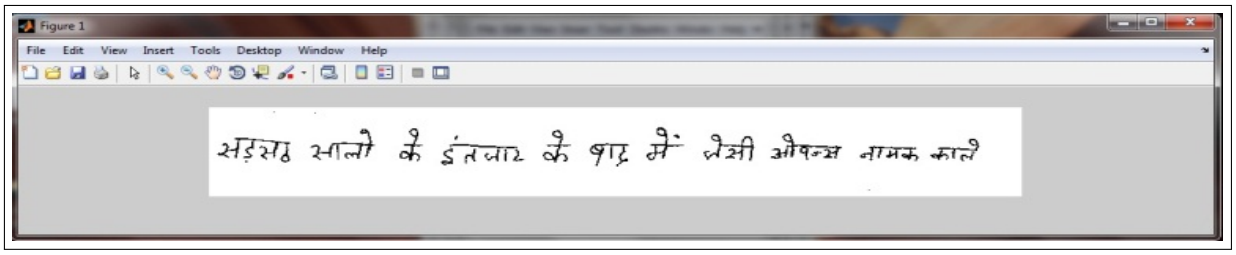


Figure 4.11: Input image to word procedure

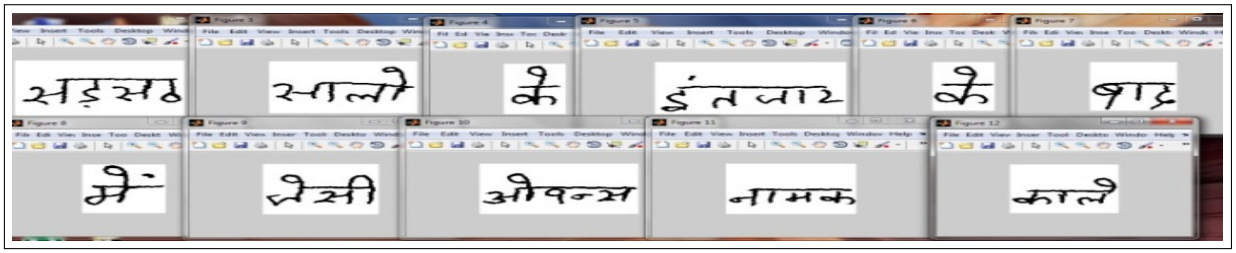


Figure 4.12: Output images from word procedure

4.1.3 Character Segmentation

This is the final stage of segmentation, which receives input from word processing function in $2-D$ array IMG_{Word} as shown in Fig. 4.13a and segment the possible characters from those words as shown in Fig. 4.13b.

(a) Input word (b) Segmented word

Figure 4.13: Input and Output of character segmentation procedure

To segment characters from word apply the following steps:

Step 1: The image height and width are read into the variable $Word_{ht}$ and $Word_{wd}$.

$$[Word_{ht}, Word_{wd}] = size(IMG_{Word}); \tag{4.1.8}$$

Step 2: Headerline as shown in Fig. 4.14 search is initiated in the word. The line is considered as headerline if it contains maximum number of black pixels and lies in the top 40% area of the $Word_{ht}$. Headerline in the word is detected using Algorithm 3.

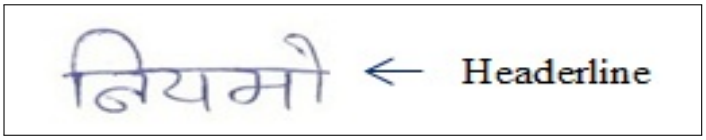


Figure 4.14: Word with header line

Algorithm 3: Header Line Detection Algorithm

Step 1: Initialize count variable to 1.

Step 2: In the top, 40% area of word height, find out the row having maximum black pixels.

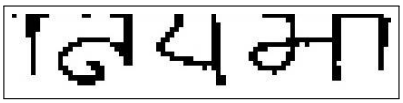
Step 3: If the selected row having black pixel count greater than 70% of the word width than it is considered as the headerline in the word otherwise there is no headerline in that word (may be numerical) .

Step 4: Initialize the headerline variable with the index of the selected row.

Step 3: After detection of headerline, remove the headerline from the word. The purpose of removing headerline is to obtained the vertical white space gap between characters.

Step 4: After removing the header line, the word is processed into two parts WD_{UPPER} as shown in Fig. 4.15a and WD_{LOWER} as shown in Fig. 4.15b. The first array WD_{UPPER} stores the upper modifier symbol above the headerline. The second array WD_{LOWER} stores the remaining data without header line. Finally, apply proposed Algorithm 4 to segment characters from words.

(a) Upper modifiers in the word

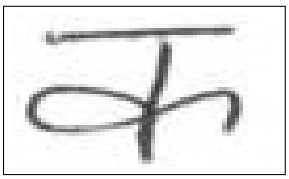


(b) Characters in the word

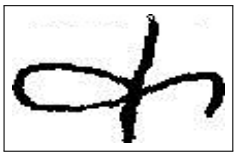
Figure 4.15: Division of word based on headerline

Demonstration of the algorithm and some of the character segmentation problems handled by the proposed algorithm are discussed in following cases:

Case 1: It is the least complex case. In it, if no upper modifiers are present in the area at the top of character and there is no pixel above the headerline, then store the character into an array as shown in Fig. 4.16.



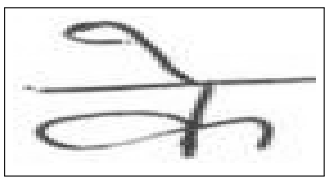
(a) Input character without upper modifiers



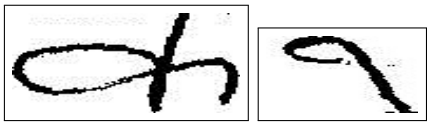
(b) Output characters after headerline removal

Figure 4.16: Character without upper modifier

Case 2: If upper modifier is present at first character but not covering the next character then segment the character and its modifier separately as shown in Fig. 4.17.



(a) Input character



(b) Segmented character

Figure 4.17: Character withnotsecondpixels upper modifier

Case 3: If upper modifier is covering both the characters from top, then two sub cases arises that are explained below:

Algorithm 4: Character Segmentation Algorithm

Input: A word from word algorithm 2
count = 1;
while $(count < Word_{wd})$ **do**
Detect possible character in WD_{LOWER} by finding starting and ending column having black pixel count is zero.
if (modifier is Present in WD_{UPPER} corresponding to detected character) **then**
Search for next character in array WD_{LOWER} by finding starting and ending column having black pixel count is zero.
if (modifier is Present in WD_{UPPER} corresponding to next detected character) **then**
if (end-to-end white pixel gap in WD_{UPPER} from start to end == true) **then**
Store (First Character);
Store (Modifier indexing with First Character);
else if (end-to-end white pixel gap in WD_{UPPER} ==false) **then**
Find the minimum height of pixels from bottom in WD_{UPPER}
if (minheight $> 20\%$ of total height) **then**
Store (First Character);
Store (Second Character);
Store (Modifier index with Second Character);
else
Find the width of both characters using end-to-end white pixels scan
if $(width1 < width2)$ **then**
Store (Attached modifier to first char);
Store (Second Character);
else
Store(Attached modifier to second character);
Store (First Character);
end if
end if
end if
else
Store (First Character);
Store (Modifier corresponding with first character);
end if
else
Store(First Character);
end if
end while

3.1 Upper modifier has two junction points with header line as shown in Fig. 4.18a and close loop area is present at top, then connect the modifier with the character having less width as shown in the Fig. 4.18 and Fig. 4.19.

(a) Width of first character is less than second character

(a) Input character (b) Segmented character

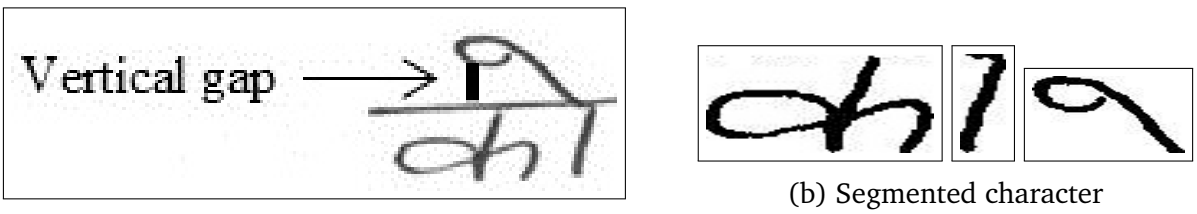
Figure 4.18: Word having closed loop in upper area

(b) Width of first character is less than second character

(a) Input character (b) Segmented character

Figure 4.19: Input character having width of second character less than other

3.2 In this case, upper modifiers having only one junction point with the header line and vertical gap is present between upper modifier and header line as shown in Fig. 4.20a. After finding the vertical gap, the characters and the modifiers are store independently as shown in Fig 4.20b.



(a) Input character (b) Segmented character

Figure 4.20: Vertical gap in upper modifier

5

Testing and Results

For experimentation, data set of 120 handwritten Hindi documents has been collected from writers of different age groups and after that these are scanned at 300 DPI on a flatbed scanner in jpg format. The sample data includes almost each type of complexity/issue discussed in Section 3. Due to limited resources and time, experimental results has been verified on 10 documents only using the proposed segmentation approach.

After applying the algorithm(s) on these documents, encouraging segmentation results has been obtained. Initially, the algorithm is applied to segment text lines with the help of average line height and white pixel gap concept. Results obtained are shown in Table 5.1.

Table 5.1: Result of line segmentation

Sample No.	**Total No. of Lines**	**Lines Correctly Segmented**	**Accuracy**
$DOC1$	18	18	100%
$DOC2$	13	13	100%
$DOC3$	12	12	100%
$DOC4$	21	21	100%
$DOC5$	10	10	100%
$DOC6$	20	20	100%
$DOC7$	18	17	94.44%
$DOC8$	17	16	94.11%
$DOC9$	15	14	93.33%
$DOC10$	16	14	87.5%
Average Accuracy			96.87%

As shown in Table 5.1, documents numbered 7-10, text lines cannot be extracted fully due to connected/overlapped lines. The accuracy variation of line segmentation function is shown in Fig. 5.1.

After line segmentation, the lines are passed to the word segmentation function.

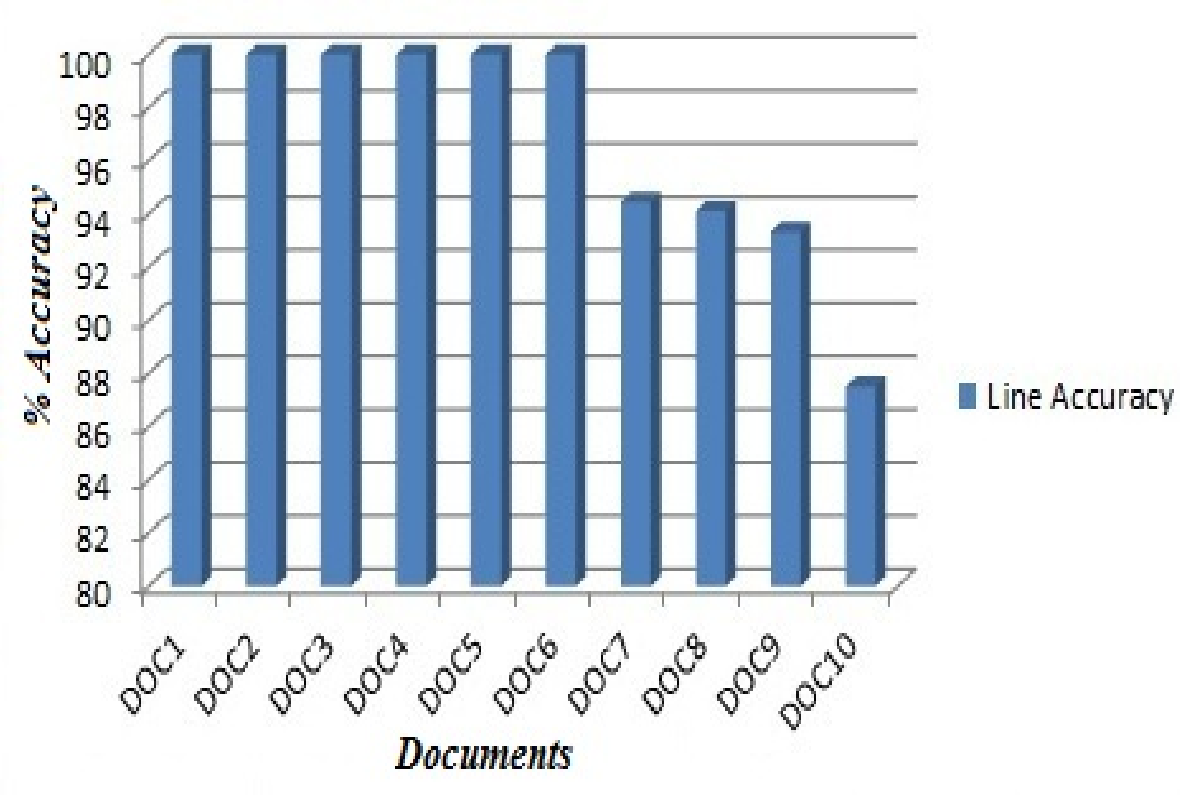


Figure 5.1: Line segmentation results obtained from proposed method

Generally, during word segmentation, the error rate is more due to white pixel gap in header line. The proposed word segmentation function efficiently handles continuous 2-3 white pixel gap in the header line of the word which increases the accuracy as shown in Table 5.2. The variation in the accuracy is mainly due to the presence of touching words

Table 5.2: Result of word segmentation

Sample No.	**Total No. of Words**	**Words Correctly Segmented**	**Accuracy**
$DOC1$	143	135	94.40%
$DOC2$	147	140	95.23%
$DOC3$	138	135	97.82%
$DOC4$	189	180	95.23%
$DOC5$	136	127	93.38%
$DOC6$	229	216	94.32%
$DOC7$	279	253	90.68%
$DOC8$	205	183	89.26%
$DOC9$	135	122	90.37%
$DOC10$	108	89	82.40%
Average Accuracy			92.45%

and white pixel gap of more than 3 pixels in the headerline as shown in Fig. 5.2.

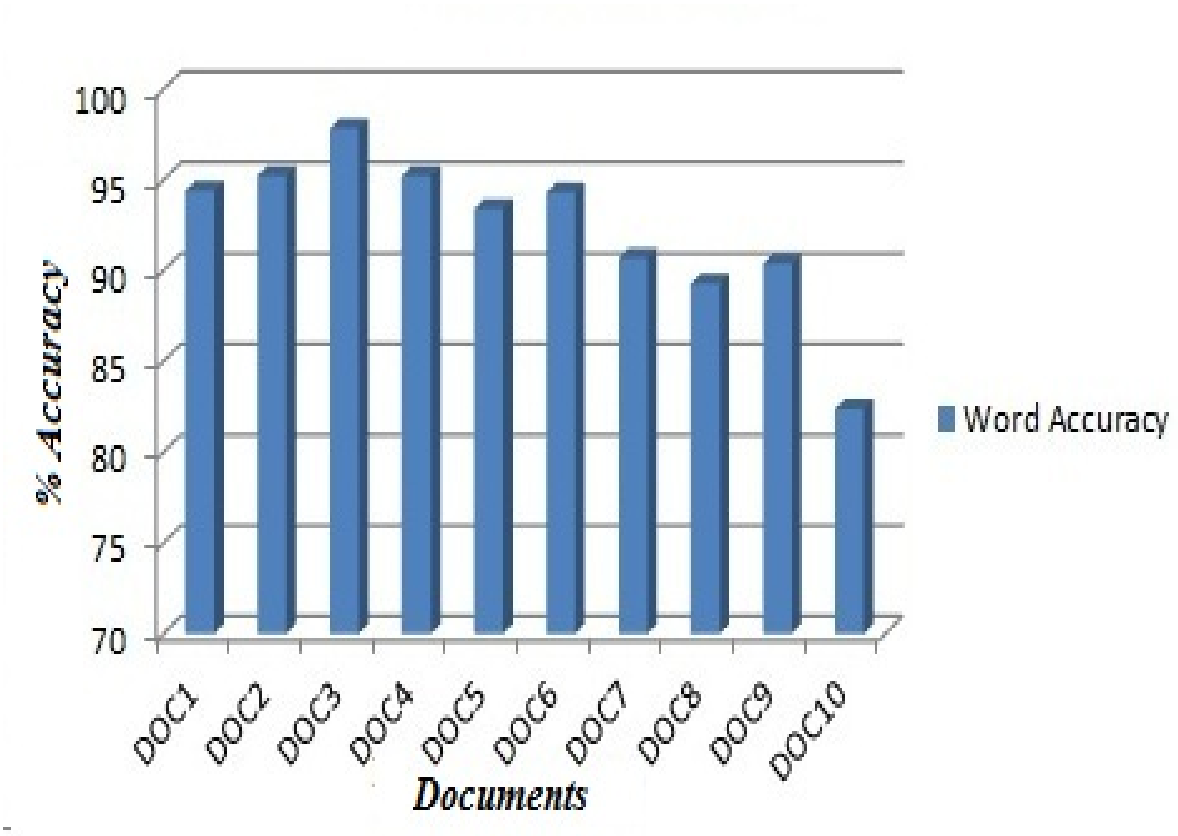


Figure 5.2: Word segmentation results obtained from proposed method

After word segmentation, the segmented words are passed to the character segmentation function. The proposed character segmentation algorithm successfully handles the upper modifiers and the characters in the words. The proposed character segmentation function gives satisfactory results which are shown in the Table 5.3.

The variation in accuracy is due to the presence of the touching modifier, touching

Table 5.3: Result of character segmentation

Sample No.	**Total No. of Characters**	**Characters Correctly Segmented**	**Accuracy**
$DOC1$	612	555	90.68%
$DOC2$	569	518	91.03%
$DOC3$	497	430	86.51%
$DOC4$	742	640	86.25%
$DOC5$	440	381	86.59%
$DOC6$	844	759	89.92%
$DOC7$	963	853	88.57%
$DOC8$	811	679	83.72%
$DOC9$	462	380	82.25%
$DOC10$	254	202	79.52%
Average Accuracy			87.13%

characters and uneven skew in the header line of the word and graph representation of these variations are shown in Fig. 5.3.

Finally, the comparison of overall accuracy of line, word and character segmentation is

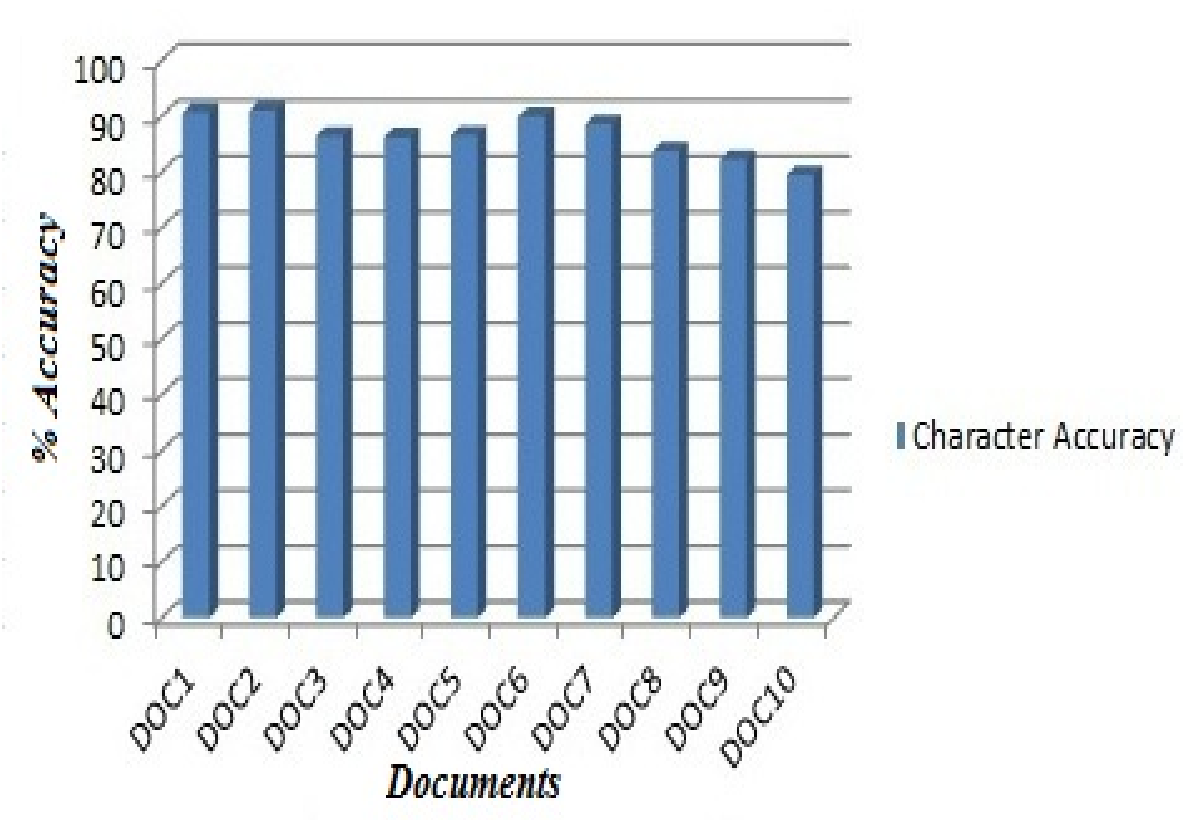


Figure 5.3: Character segmentation results obtained from proposed method

represented using Fig. 5.4 which clearly represent accuracy of the lower phases with upper phases. After detailed analysis and study it is explored that one has to give more attention to the upper phases too achieve higher accuracy rate. The less number of errors in the upper phase result into more accuracy and more number of errors in the upper phase will leads to less accuracy. The comparison of the accuracy of proposed character segmentation function with existing work is shown in Table 5.4 and represent graphically using Fig. 5.5.

Table 5.4: Comparison of proposed method with existing techniques

SR. No	Ref.	Accuracy on character level
1	Veena et. al [33]	83%
2	Garg et. al [34]	85.74%
3	**Proposed Technique**	**87.13 %**

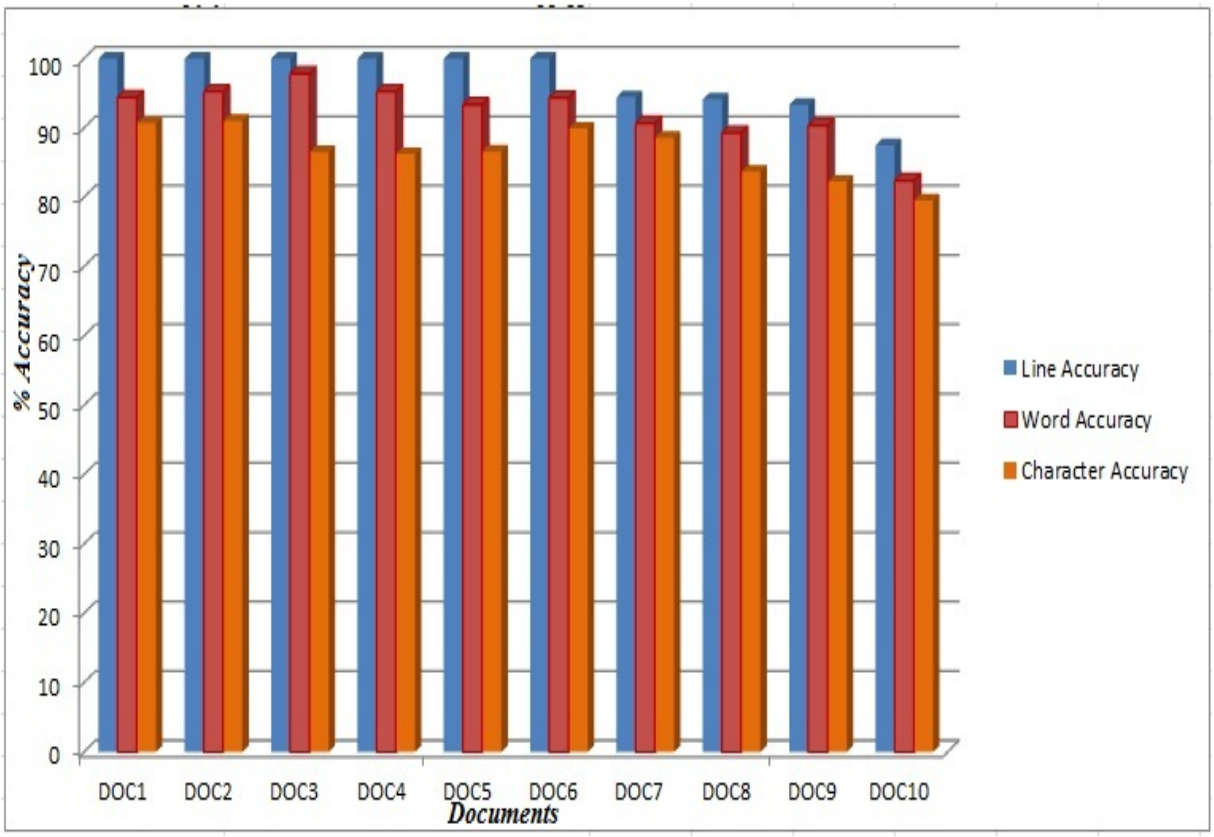


Figure 5.4: Accuracy at various stages of segmentation

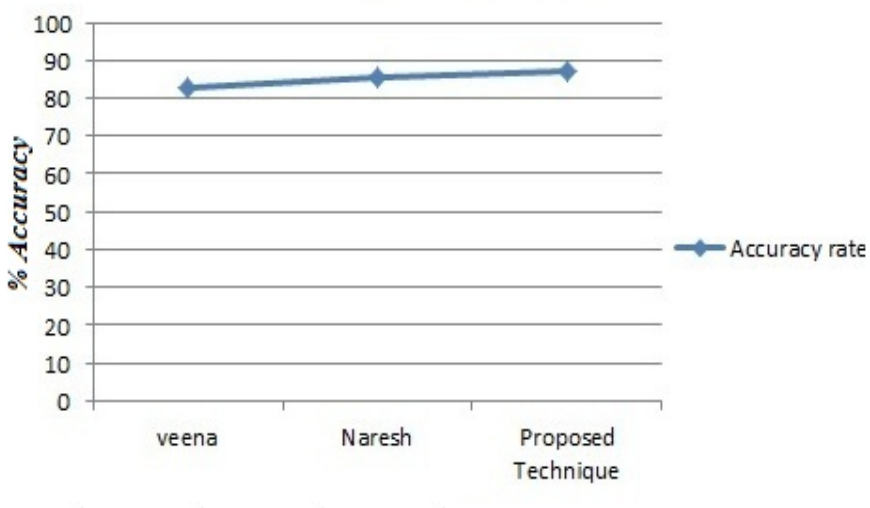


Figure 5.5: Comparison of proposed technique with existing techniques

6

Conclusions and Future Work

6.1 Conclusions

In this dissertation, segmentation stage of OCR system on handwritten Hindi documents has been explored. The major contribution of the thesis are:

1. Line segmentation algorithm is proposed, which successfully handle the connected and overlapped lines by using the average line height approach. This has been achieved by clustering the document into three columns which gives more accurate line height results.

2. The degradation (upto 2-3 consecutive missing black pixels) in the headerline of the word has also been handled successfully using the proposed word segmentation algorithm. The headerline of the word gives estimation of the orientation of characters in the word. There is high probability that the headerline can leads to ambiguity. It can be handled by aligning the characters along with the headerline. This will increase the overall accuracy of the system.

3. The proposed character segmentation algorithm successfully segment the characters and modifiers with an accuracy rate of 87.13%. The accuracy is effected due to the compound and fused characters.

6.2 Future Research

Researchers can work upon the way to segment compound and fused character symbols to obtain more accurate results. Also, handwritten analysis system can be developed to identify writer/person. Presently, the algorithm has been tested only on handwritten Hindi documents. It can be extended for other languages like Gurumukhi, Marathi, Sanskrit *etc.*

References

[1] M. Meshesha and C. V. Jawahar, "Optical character recognition of amharic documents." *African J. of Inf. & Commun. Technology*, vol. 3, no. 2, 2007. [Online]. Available: http://dblp.uni-trier.de/db/journals/ajict/ajict3.html#MesheshaJ07

[2] G. Nagy, "Twenty years of document image analysis in pami," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 1, pp. 38–62, Jan 2000.

[3] R. Jayadevan, S. Kolhe, P. Patil, and U. Pal, "Offline recognition of devanagari script: A survey," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 41, no. 6, pp. 782–796, Nov 2011.

[4] P. Krishnan, R. Shekhar, and C. V. Jawahar, "Content level access to digital library of india pages," in *Proceedings of the Eighth Indian Conference on Computer Vision, Graphics and Image Processing*, ser. ICVGIP '12. New York, NY, USA: ACM, 2012, pp. 5:1–5:8. [Online]. Available: http://doi.acm.org/10.1145/2425333.2425338

[5] N. Otsu, "A threshold selection method from gray-level histograms„" *IEEE Transactions on Systems, Man, and Cybernetics, Vol. 9, No. 1,*, 1979.

[6] Y. Bassil and M. Alwani, "Ocr post-processing error correction algorithm using google online spelling suggestion," *arXiv preprint arXiv:1204.0191*, 2012.

[7] I. K. Sethi and B. Chatterjee, "Machine recognition of hand printed devanagari numerals," *J. Inst. Electron. Telecommun. Eng., vol. 22, pp. 532Ű535*, 1977.

[8] N. Garg, L. Kaur, and M. Jindal, "A new method for line segmentation of handwritten hindi text," in *Information Technology: New Generations (ITNG), 2010 Seventh International Conference on*, April 2010, pp. 392–397.

[9] B. Chaudhuri and S. Bera, "Handwritten text line identification in indian scripts," in *Document Analysis and Recognition, 2009. ICDAR '09. 10th International Conference on*, July 2009, pp. 636–640.

[10] R. Kumar and A. Singh, "Detection and segmentation of lines and words in gurmukhi handwritten text," in *Advance Computing Conference (IACC), 2010 IEEE 2nd International*, Feb 2010, pp. 353–356.

[11] Y.-K. Chen and J.-F. Wang, "Segmentation of single- or multiple-touching handwritten numeral string using background and foreground analysis," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 11, pp. 1304–1317, Nov 2000.

[12] N. Arica and F. Yarman-Vural, "An overview of character recognition focused on off-line handwriting," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 31, no. 2, pp. 216–233, May 2001.

[13] P. Slavik and V. Govindaraju, "Equivalence of different methods for slant and skew corrections in word recognition applications," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 23, no. 3, pp. 323–326, Mar 2001.

[14] U. Garain and B. Chaudhuri, "Segmentation of touching characters in printed devnagari and bangla scripts using fuzzy multifactorial analysis," in *Document Analysis and Recognition, 2001. Proceedings. Sixth International Conference on*, 2001, pp. 805–809.

[15] Z. Shi and V. Govindaraju, "Line separation for complex document images using fuzzy runlength," in *null*. IEEE, 2004, p. 306.

[16] R. Manmatha and J. L. Rothfeder, "A scale space approach for automatically segmenting words from historical handwritten documents," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 8, pp. 1212–1225, Aug 2005.

[17] M. Jindal, R. Sharma, and G. Lehal, "Segmentation of horizontally overlapping lines in printed gurmukhi script," in *Advanced Computing and Communications, 2006. ADCOM 2006. International Conference on*. IEEE, 2006, pp. 226–229.

[18] A. Zahour, B. Taconet, L. Likforman-Sulem, and W. Boussellaa, "Overlapping and multi-touching text-line segmentation by block covering analysis," *Pattern analysis and applications*, vol. 12, no. 4, pp. 335–351, 2009.

[19] G. Louloudis, B. Gatos, I. Pratikakis, and C. Halatsis, "Text line detection in handwritten documents," *Pattern Recognition*, vol. 41, no. 12, pp. 3758–3772, 2008.

[20] N. Stamatopoulos, B. Gatos, and S. J. Perantonis, "A method for combining complementary techniques for document image segmentation," *Pattern Recognition*, vol. 42, no. 12, pp. 3158–3168, 2009.

[21] Y. Li, Y. Zheng, D. Doermann, S. Jaeger, and Y. Li, "Script-independent text line segmentation in freestyle handwritten documents," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 30, no. 8, pp. 1313–1329, Aug 2008.

[22] A. Nicolaou and B. Gatos, "Handwritten text line segmentation by shredding text into its lines," in *Document Analysis and Recognition, 2009. ICDAR '09. 10th International Conference on*, July 2009, pp. 626–630.

[23] A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 5, pp. 855–868, May 2009.

[24] A. Pezeshk and R. Tutwiler, "Automatic feature extraction and text recognition from scanned topographic maps," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 49, no. 12, pp. 5047–5063, Dec 2011.

[25] X. H. Yi-Feng Pan and C.-L. Liu, "A hybrid approach to detect and localize texts in natural scene images," *IEEE Transactions on Image Processing, vol. 20, no. 3*, March 2011.

[26] A. A. Gulhane and A. S. Alvi, "Noise reduction of an image by using function approximation techniques," *International Journal of Soft Computing and Engineering (IJSCE) Volume-2, Issue-1*, 2012.

[27] D. Salvi, J. Zhou, J. Waggoner, and S. Wang, "Handwritten text segmentation using average longest path algorithm," in *Applications of Computer Vision (WACV), 2013 IEEE Workshop on*, Jan 2013, pp. 505–512.

[28] G. Pirlo and D. Impedovo, "Fuzzy-zoning-based classification for handwritten characters," *Fuzzy Systems, IEEE Transactions on*, vol. 19, no. 4, pp. 780–785, Aug 2011.

[29] S. Shelke and S. Apte, "A fuzzy based classification scheme for unconstrained handwritten devanagari character recognition," in *Communication, Information Computing Technology (ICCICT), 2015 International Conference on*, Jan 2015, pp. 1–6.

[30] M. Kumar, M. Jindal, and R. Sharma, "A novel hierarchical technique for offline handwritten gurmukhi character recognition," *National Academy Science Letters*, vol. 37, no. 6, pp. 567–572, 2014.

[31] S. Singh, T. Kariveda, J. Gupta, and K. Bhattacharya, "Handwritten words recognition for legal amounts of bank cheques in english script," in *Advances in Pattern Recognition (ICAPR), 2015 Eighth International Conference on*, Jan 2015, pp. 1–5.

[32] A. Gaur and S. Yadav, "Handwritten hindi character recognition using k-means clustering and svm," in *Emerging Trends and Technologies in Libraries and Information Services (ETTLIS), 2015 4th International Symposium on*, Jan 2015, pp. 65–70.

[33] V. Bansal and R. Sinha, "Segmentation of touching characters in devanagari," IIT, Kanpur, India, Tech. Rep., 2010.

[34] N. K. Garg, L. Kaur, and M. Jindal, "Segmentation of handwritten hindi text," *International Journal of Computer Applications (IJCA)*, vol. 1, no. 4, pp. 22–26, 2010.

Communicated Papers

Detail of communicated article is given below:

- ⋆ R. Mittal and K. Jindal, "A New Method For Segmenting Characters and Modifiers Of Handwritten Devanagari Documents", *IEEE International Conference on CGVIS 2015*, **Under Review**.

Video Presentation

Below is the link of my video presentation which you can watch on YouTube.

http://youtu.be/ypqmd2xso5k.