

Writer's Behavior Identification Based on Online Graphometric Parameters

By
Dhruba Joyti Bhuiya
Kuheli Pratihar
Supriti Paul
Urmi Halder

UNDER THE GUIDANCE OF
Mr. Rajib Saha

**PROJECT REPORT SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF**

**BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND
ENGINEERING**

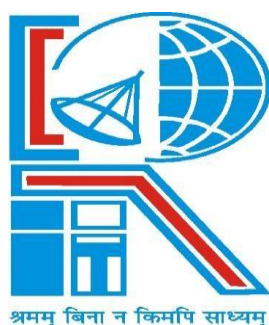
RCC INSTITUTE OF INFORMATION TECHNOLOGY

Session 2017-2018



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
RCC INSTITUTE OF INFORMATION TECHNOLOGY
[Affiliated to West Bengal University of Technology]
CANAL SOUTH ROAD, BELIAGHATA, KOLKATA-700015

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
RCC INSTITUTE OF INFORMATION TECHNOLOGY**



TO WHOM IT MAY CONCERN

I hereby recommend that the Project entitled **Writer's Behavior Identification Based on Online Graphometric Parameters** prepared under my supervision by **Dhruba Joyti Bhuiya** (Reg. No. 141170110029 of 2014-2015, Class Roll No. CSE2014/003), **Kuheli Pratihar** (Reg. No. 141170110037 of 2014-2015, Class Roll No. CSE2014/051), **Supriti Paul** (Reg. No. 141170110083 of 2014-2015, Class Roll No. CSE2014/029), **Urmi Halder** (Reg. No. 141170110093 of 2014-2015, Class Roll No. CSE2014/007) of B.Tech (7th /8th Semester), may be accepted in partial fulfillment for the degree of **Bachelor of Technology in Computer Science & Engineering** under West Bengal University of Technology(WBUT).

.....
Project Supervisor
Department of Computer Science and Engineering
RCC Institute of Information Technology

Countersigned:

.....
Head
Department of Computer Sc. & Engg,
RCC Institute of Information Technology
Kolkata – 700015.

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
RCC INSTITUTE OF INFORMATION TECHNOLOGY**



CERTIFICATE OF APPROVAL

The foregoing Project is hereby accepted as a credible study of an engineering subject carried out and presented in a manner satisfactory to warrant its acceptance as a prerequisite to the degree for which it has been submitted. It is understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein, but approve the project only for the purpose for which it is submitted.

FINAL EXAMINATION FOR
EVALUATION OF PROJECT

1. _____

2. _____

(Signature of Examiners)

ACKNOWLEDGEMENT

The satisfaction that accompanies the progress of this work would be incomplete without the mention of the people who made it possible, without whose constant guidance and encouragement would have made efforts go in vain. I consider myself privileged to express gratitude and respect towards all those who guided us through the considerable progress of this project.

We convey thanks to our guide Mr. RAJIB SAHA of Computer Science and Engineering Department for providing encouragement, constant support and guidance which was of a great help and motivation to us.

Table of Contents

	Page No.
1. Introduction	1
2. Review of Literature	2
3. Objective of the Project.....	5
4. System Design.....	7
5. Methodology for implementation (Formulation/Algorithm)	8
6. Implementation Details.....	13
7. Results/Sample output.....	24
8. Conclusion.....	33

References

Appendix-: Program Source code with adequate comments.

1. Introduction

Handwriting analysis is described as a scientific study of human behavior through handwriting. The scientific name for handwriting analysis is Graphology. It is a multistage procedure. It begins with collecting the handwriting samples on plain white A4 size paper. Handwriting analysis needs to perform pre-processing steps such as conversion to grayscale, binarization etc. for better recognition. Initially scanned handwriting image is taken as an input then thresholding is done to convert the image into binary image.

But offline handwriting analysis suffers from several shortcomings like determination of the writing angle and speed.

Thus, to get more accurate result we have done online handwriting analysis using a device called Take Note with storage capability that digitally captures and stores everything instantly one writes or draws with ink on ordinary paper when connected to a PC. Then, one can easily view, edit, organize and share handwritten notes in Windows. In other words, Take Note offers an online writing function which can instantly synchronize one's writing on the paper with the digital page in its software in Windows.

We are developing a local approach, based on the extraction of characteristics that are specific to a writer. To exploit the existence of redundant patterns within a handwriting sample, the writing is divided into a large number of small sub-images by means of line segmentation, word segmentation and character segmentation. Graphologists predict the personality of the writer with a piece of handwriting. Not only the accuracy of the handwriting analysis depends on the skills of the graphologists but also the manual process is costly and prone to fatigue.

Hence the proposed methodology focuses on developing a tool for online writer identification which can predict the behavior of the writer automatically with the help of a computer from the following parameters: -

- Writing Speed
- Under Length and Upper Length of some characters
- Number and Type of Strokes
- Completeness of characters

In order to predict the behavior with reference to the above-mentioned parameters, we shall refer to a database, authenticated by Graphologist and Psychologist.

2. Review of Literature

This section surveys work in character segmentation and character analysis of the writer from handwritten document images. The subsequent techniques either achieved remarkable results in the corresponding test datasets, or are incorporated in the workflows of integrated systems for specific tasks.

In 2016, Abhishek Bal and Rajib Saha ^[1] proposed an off-line handwritten document analysis through segmentation, skew recognition and writing pressure for cursive handwritten document. The proposed segmentation method is based on modified horizontal and vertical projection that can segment the text lines and words even if the presence of overlapped and multi-skewed text lines. Proposed work also present orthogonal projection based baseline recognition and normalization method as well as writing pressure detection method that can predict the personality of writer from the baseline and writing pressure.

In the proposed method, after creating the horizontal projection histogram of a binary document image, count the number of rising section and height of each rising section. The average height of the rising sections is treated as the threshold. Then consider each and every rising section and check the height of that rising section is greater than or equals to the threshold or not. If yes then based on that rising section of the horizontal histogram, the line was individually segmented from the actual binary document image, otherwise neglect that rising section as a false line segment. These types of the false rising section may occur due to overlapping between two lines or presence of a bar in an upper letter. The most important are that when a rising section is treated as a false line segment and next rising section is treated as a true line segment then the portion of the false line segment is added to true line segment for the segmentation of next line from the actual document image, otherwise, some features are removed.

After line segmentation, it may happen different lines may have the different skew angle. The proposed skew normalization process is based on orthogonal projection length.

In the case of word segmentation, to segment, the words from the line, firstly inter-word and intra-word gaps are measured. Inter-word gaps denote the gaps between two words and intra-word gaps denote gaps within a word. Generally, gaps between the words are larger than the gaps within a word. These proposed methods construct the vertical projection histogram to measure the width of each inter-word and intra-word gaps then it measures the threshold value to differentiate between inter-word and intra-word gaps. If the width of gaps is greater than or equals to threshold then gaps are treated as inter-word gaps and words are segmented individually from the line depending on the threshold.

If a line has global skew then it may possible that several words within a line may have different skew. So it may require normalizing the skew of the words for a single line. For that reason again proposed skew normalization method is applied to the each segmented word separately.

In 1995, Nelson H C Yung, Andrew H S Lai* and Perry ZP Chua ^[3] proposed a new character segmentation algorithm for dealing with off-line handwritten script recognition. The X-axis projection, Y-axis projection and geometric classes techniques used by the algorithm proves to be successful in segmenting normal handwriting with a success rate of 93 .5%. Because of this development, detailed understanding of geometric classes of English characters and the difficult cases in segmentation was gained.

In this paper, a character segmentation algorithm is presented for off-line handwritten script recognition. This work is driven by the believe that a good and accurate character segmentation process will greatly improve the recognition rate of the system. This is contrary to leaving the problem to a later stage, relying on the classifier and/or spell checker/lexicon, as adopted by some other groups. The segmentation algorithm described here relies on three principles: X-axis projection (vertical), Y-axis projection (horizontal) and geometric classes of characters and words. By employing these three principles appropriately in the algorithm, minimum width characters can be segmented out without much problem.

Apart from requiring the written sample to have a consistent writing style, words and lines are separated by detectable gaps, and lines should be near horizontal, all the other normal variations in handwriting are allowed and can be accommodated by the algorithm. In essence, the use of Y-axis projection in conjunction with the concept of geometric classes improves the X-axis projection results and fine tune the final list of segmentation points. The detailed analysis of the segmentation results shows that there exist characters such as

'n', 'm', 'u', 'w' that are naturally difficult to be segmented. Fortunately, the occurrence of some of these characters in a page of text is not high compared with some others such as 'e', 'i', 's'. Based on this argument, normal segmentation success rate is expected to be well above 90%.

In 2012, Parmeet Kaur Grewal, Deepak Prashar^[2] proposed a method to predict the behavior of a person from the baseline, the letter slant, pen pressure, letter 'i' and letter 'f'. These parameters are input to the Artificial Neural Network which predicts the behavior of the writer.

In this paper, a method has been proposed to predict the behavior of a person from the features extracted from his handwriting. The personality traits revealed by baseline, letter slant, pen pressure, letter 'i' and letter 'f' as found in individual's handwriting are explored in this paper. Five parameters, baseline, slant, pen pressure, letter 'i' and letter 'f' are input to the ANN which outputs the personality trait of the writer. The evaluation of the baseline and letter slant is using the polygonization method, the evaluation of pen pressure utilizes grey-level threshold value, and evaluations of letter 'i' and letter 'f' use template matching. MATLAB is the tool used for this purpose. The performance is measured by examining multiple samples.

In 1988, C.C. Tappert, C.Y. Suen, T. Wakahara^[4] described the state-of-the-art of on-line handwriting recognition during a period of renewed activity in the field. Shape recognition algorithms, preprocessing and post-processing technique, experimental systems, and commercial products are examined.

In 2000, R. Plamondon, S.N. Srihari^[5] described algorithms for preprocessing, character and word recognition, and performance with practical systems are indicated. Other fields of application, like signature verification, writer authentication, handwriting learning tools are also considered.

In June 2015 thesis submitted by Rohit Mittal under the supervision of Mr. Khushneet Jindal^[6] developed algorithms for line, word and character segmentation for handwritten Hindi documents. 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, over lapped text lines, connected components between lines, unequal height of characters, separation of upper and lower modifiers.

The primary objectives of this dissertation is to be evolved an offline handwritten character recognition system for Hindi language. The present study represent the ways in which the system can recognize the offline handwritten characters written by different writers.

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 in 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.

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. A character segmentation has been proposed which successfully handles the over segmentation problems.

In 2017, Pritam Dhande and Reena Kharat^[7] represented the work related to recognition of cursive English handwriting. Character recognition of handwritten cursive English script is a very challenging task. In cursive English handwriting, the characters in a word are connected to each other. So the segmentation and feature extraction of cursive English script is much difficult. In the proposed work, horizontal and vertical projection

methods are used for segmentation. Convex hull algorithm is used for feature extraction and SVM is used for classification and recognition.

Optical character recognition is also called as optical character reader and it is abbreviated as OCR. OCR translates the images into machine readable format such as ASCII or Unicode. Character recognition can be classified into two types based on the type of the text i.e. machine printed text and handwritten text. When a printed text is converted to machine readable text then we can search through it with keywords, compress, edit and send it, and can store in much less space.

The main challenge in the recognition of handwritten characters is that every person on the earth has different handwriting. There are various other factors also which causes difference in handwriting such as multi-orientations, skew of the text lines, overlapping characters, connected components, pressure points etc. Many scripts are there with their intrinsic variations. A single character can be written in many forms, so it is a challenging task to recognize a particular handwritten character.

The link mentioned in ^[8] provides us different thinning algorithms.

Thinning algorithm is a Morphological operation that is used to remove selected foreground pixels from binary images. It preserves the topology (extent and connectivity) of the original region while throwing away most of the original foreground pixels.

Thinning is somewhat like erosion or opening. It can be used for several applications, but is particularly useful for skeletonization and Medial Axis Transform. In this mode it is commonly used to tidy up the output of edge detectors by reducing all lines to single pixel thickness. Like other morphological operators, thinning operators take two pieces of data as input. One is the input image, which maybe either binary or grey scale. The other is the structuring element, which determines the precise details of the effect of the operator on the image.

Thinning algorithms can be divided into two broad classes namely iterative and non-iterative. Almost all iterative thinning algorithms use Mark-and-Delete templates including Stentiford Thinning Method. Both Stentiford and Zhang-Suen methods use Connectivity numbers to mark and delete pixels. There is a fast non-iterative thinning algorithm proposed by Neusius-Olszewski. The Connectivity number is a measure of how many objects are connected with a particular pixel.

Alberto Martin and Sabri Tosunoglu^[9] analyzed different Image Processing Algorithms by classifying them in logical groups. In addition, specific methods are presented illustrating the application of such techniques to the real-world images. In most cases more than one methods are used. This allows a basis for comparison of different methods as advantageous features as well as negative characteristics of each technique is delineated.

The Image Algebra forms a solid theoretical foundation to implement computer vision and image processing algorithms. With the use of very efficient and reliable high-level computer languages such as C/C++, Fortran 90, and Java, innumerable image processing and machine vision algorithms have been written and optimized. All this code written and compiled has become a powerful tool available for researchers, scientists and engineers, which further accelerated the investigation process and incremented the accuracy of the final results. The discussion of the Basic Machine Vision and Image Processing Algorithms should be divided in five major groups

- Grey-Level Segmentation or Thresholding Methods
- Edge-Detection Techniques
- Digital Morphology
 - Texture
- Thinning and Skeletonization Algorithms

Skeletonization was introduced to describe the global properties of objects and to reduce the original image into a more compact representation. The skeleton expresses the structural connectivities of the main components of an object and it has the width of one pixel in the discrete case. These kinds of techniques have a wide range of applications, for example skeletonization has been applied successfully in solving character recognition problems.

A basic method for skeletonization is thinning. It is an iterative technique, which extracts the skeleton of an object as a result. In every iteration, the edge pixels having at least one adjacent background point are

deleted. All those pixels can be eroded, only if its removal doesn't affect the topology of the object. The skeleton represents the shape of the object in a relatively small number of pixels .

Thinning works for objects consisting of lines (straight or curved). This method does not work for object having shapes that encloses a large area. Thinning is most of the time an intermediate process, to prepare the object for further analysis. The subsequence processes determine the properties of the skeleton. For the same object, a skeleton may work find in one situation, but may not work in all situations.

In 2013, Ashwini S. Karne^{a*} and S. S. Navalgund^{a [10]} implemented image thinning operation on a binary image of 128 x 128 pixels using Zhang Suen's thinning algorithm. The proposed work is designed using MATLAB 7.12 and also synthesized by mapping on Virtex 5 in Xilinx ISE for understanding the hardware complexity. Simulation results are obtained in terms of waveforms in ISim Xilinx ISE Simulator and the output text file of the hardware system is converted to an image format using MATLAB.

Thinning is a process of extracting a skeleton from an object in a digital image. It can also be defined as act of identifying those pixels belonging to an object that are essential for communicating the object's shape: these are the skeletal pixels, and form a set. Thinning provides a convenient and condensed representation of image object information. Skeleton of an object can preserve topological properties, reduce storage requirements and reduce the transmission time. Thinning is also termed as skeletonization. Skeletonization is widely used in many image pre-processing applications, such as character recognition, pattern recognition, image coding and biological shape description. The proposed work focuses on extracting the centre line of a binary image using Zhang-Suen thinning algorithm. Performance measurement is carried out between Zhang – Suen's thinning algorithm and MATLAB command for image thinning in terms of Thinning Rate (TR).

The result of skeletonization using thinning algorithms must have the following properties: As thin as possible, Connected and Centred.

The link mentioned in ^[11] provide us the basic approach of template matching, template matching using correlation, problems associated, normalized correlation and finding matches.

A template is a small image (sub-image) .The goal is to find occurrences of this template in a larger image. That is, you want to find matches of this template in the image.

If the image intensity varies with position, Correlation can fail. – For example, the correlation between the template and an exactly matched region can be less than correlation between the template and a bright spot. The range of $c(x,y)$ is dependent on the size of the feature. Correlation is not invariant to changes in image intensity – Such as lighting conditions.

We can normalize for the effects of changing intensity and the template size. Normalized Correlation is robust – It is one of the most commonly used template matching criteria when accuracy is important. But, it is computationally expensive. For speed, we often use other similarity metrics.

3. Objective of the project

The goal of our project is to predict a writer's behavior based on the selected graphometric parameters; writing speed, under length and upper length of some characters, starting and terminal strokes, completeness of characters. In order to predict the behavior with reference to the above-mentioned parameters, we shall refer to a database, authenticated by Graphologist and Psychologist. We are developing a local approach, based on the extraction of characteristics that are specific to a writer.

4. System Design

4.1 Sample Collection

A handwritten test document is taken as input along with the time elapsed to complete the document. A device is used to take the input as it is being written for online analysis.

4.2 Image Pre-processing

Image pre-processing^[1] is the technique in which the handwritten sample is translated into a format which can be easily and efficiently processed in further steps. These steps involve binarization, noise removal, line segmentation, word segmentation and skew normalization. Binarization converts gray scale image into binary image. Quality of the converted image is improved by applying noise removal techniques.

4.3 Line Segmentation

After binarization and noise removal, the converted image is processed through a line segmentation^[1] technique to split the hand-written document into separate lines based on rising section of the horizontal projection histogram of document image.

In English handwriting^[3] document image, most of the time no gaps are present between two lines, which may create incorrect line segmentation due to overlapping between two lines if simple horizontal projection histogram is concern. After creating the horizontal projection histogram of a binary document image, count the number of rising section and height of each rising section. The average height of the rising sections is treated as the threshold. Then consider each and every rising section and check the height of that rising section is greater than or equals to the threshold or not. If yes then based on that rising section of the horizontal histogram, the line was individually segmented from the actual binary document image, otherwise neglect that rising section as a false line segment. These types of the false rising section may occur due to overlapping between two lines or presence of a bar in an upper letter. The most important are that when a rising section is treated as a false line segment and next rising section is treated as a true line segment then the portion of the false line segment is added to true line segment for the segmentation of next line from the actual document image, otherwise, some features are removed.

4.4 Word Segmentation

In the case of word segmentation^[1], to segment, the words from the line, firstly inter-word and intra-word gaps are measured. Inter-word gaps denote the gaps between two words and intra-word gaps denote gaps within a word. Generally, gaps between the words are larger than the gaps within a word. These proposed methods construct the vertical projection histogram to measure the width of each inter-word and intra-word gaps then it measures the threshold value to differentiate between inter-word and intra-word gaps. If the width of gaps is greater than or equals to threshold then gaps are treated as inter-word gaps and words are segmented individually from the line depending on the threshold.

If a line has global skew, then it may possible that several words within a line may have different skew. So, it may require normalizing the skew of the words for a single line. For that reason, again skew normalization method is applied to each segmented word separately.

4.5 Character Extraction

After words are separated, character segmentation^[3] needs to be carried out to extract the individual characters from the handwritten script^[3].

4.6 Template Matching

Then, extracted characters are to be matched with an image dataset and recognized.

5. Methodology for implementation

5.1 Preprocessing

Binarization and Noise Removal

A binary image is a digital image that has only two possible values for each pixel. Typically, the two colors used for a binary image are black and white.

Binarization is the process of converting a pixel image to a binary image.

The RGB image is converted to gray scale using MATLAB function 'rgb2gray()' .

The gray scale image is converted to binary image using MATLAB function 'imbinarize()' .

Noise (small dots or foreground components) may be introduced easily into an image while scanning the handwritten word image during image acquisition, it is very necessary to eliminate the noise from the word images so as to make the word images fit for further processing.

MATLAB's medfilt2(A) is used for noise removing technique which performs median filtering of the image A in two dimensions.

Each output pixel contains the median value in a 3-by-3 neighborhood around the corresponding pixel in the input image.

medfilt2 pads the image with 0s on the edges, so the median values for points within one-half the width of the neighborhood ($\lfloor m/2 \rfloor$) of the edges might appear distorted.

5.2 Line Segmentation

The proposed algorithm^[6] detects the text lines in the document and successfully handle almost all the problems of line segmentation.

Assumption:

Work has already been done on skew normalization^[1] .

We are using handwritten document with zero or minimum skew i.e. the lines are almost horizontal and there is distinct space between two lines.

Algorithm:

1. Initially, Image is loaded into an image array named IMG[][][] or IMG[][]; then the image height and width are read into the variables 'rows', 'columns'.
2. Depending on the number of color channels, the image is converted into Grayscale, if required. The resulting image is then pre-processed to remove anomalies like noise. i.e. IMG[][]
3. The resultant image obtained is now converted to binary form using the medfilt2(IMG) function in MATLAB. i.e. IMG[][]
4. To detect the line, inter-line space divide the image into three equal column arrays C1,C2,C3, then each column has been searched for clear end-to-end white space gap to detect the lines in particular column.

Now, find the average line height of these three columns independently using the equation:

$$CL_i = \sum_{j=1}^M (CL_{i(L(j)}) / M_i \quad (1)$$

Where $i=1,2,3$

$j=1,2, 3 \dots ,M$

M_i =number of lines detected in the column array C_i

Then find the average inter-line space gap of these three columns independently using the equation:

$$CS_j = \sum_{i=1}^N (CS_{i(ILG(j)}) / N_i \quad (2)$$

Where $i=1,2,3$

$j=1,2, 3 \dots ,N$

N_i =number of inter-line gaps detected in the column array C_i .

For more accurate segmentation result:

Average line-height= $CL_1+CL_2+CL_3/3$

Average inter-line gap= $CS_1+CS_2+CS_3/3$

5. After determining the Average line-height and the Average inter-line gap, the *start* variable is set to the starting location of the pre-processed image
6. Detect the possible line by finding the starting and ending row having end-to-end black pixel count as zero.
7. If the line height of the detected line is less than the average line height, then go to step 13 else go to step 8.
8. 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.
9. Find the (Average inter-line gap/Average line-height) % area above and below the temporary line-cut.
10. In that area, find out the row having minimum number of black pixels.
11. If the minimum black pixels in that row are less than 5%, then make the temporary line-cut as permanent line-cut and go to step 12 else go to step 13.
12. Pass the line to the Word Segmentation function from start index of the line to the permanent line cut and increment the *start* variable with the line height.
13. Pass the line to Word Segmentation function for further processing and increment the *start* variable with the line height.
14. If end of page, then stop else go to step 6(for next possible line).

5.3 Word Segmentation

For word segmentation, first inter-word and intra-word distance is calculated. Vertical projection histogram calculates the threshold value. If the distance is less or equal to the threshold value then it is considered as intra-word gap. Vertical projection method is used for word segmentation. ^[1]

Algorithm:

1. Read a segmented binary line as 2-d binary image LN[][].
2. Construct the vertical projection histogram of the line LN[][] and store into a 2-d array LVP[][].
3. From the vertical projection histogram (LVP[][]), measure width of each inter-word and intra-word gaps and store the width into 1-d array GAPSW[].
4. Count total number gaps as TGP by calculating the size of GAPSW[]. Add width of all gaps by adding the elements of GAPSW[] and store into TWD.
5. Calculate the threshold (Ti) as follows:

$$T_i = TWD / TGP \quad (3)$$

Where, T_i is the threshold value denoting average width of inter-word gaps, TWD denotes total width of all gaps and TGP denotes the total number of gaps.

6. For each $i(1 \leq i \leq \text{sizeof}(GAPSW[]))$, if $GAPSW[i] \geq T_i$ then this gaps is treated as inter-word gaps, otherwise gaps is treated as an intra-word gaps. Depending on inter-word gaps width, words are segmented from the line.
7. End

5.4 Thinning

In pre-processing steps of character segmentation one of the requirement is thinning of the binarized image. Thinning is required Convert binary shapes obtained from edge/boundary detection or thresholding to 1-pixel wide lines for better representation and further processing.

Thinning algorithm is a Morphological operation that is used to remove selected foreground pixels from binary images. It preserves the topology (extent and connectivity) of the original region while throwing away most of the original foreground pixels ^[8]. This type of thinning processes uses templates, where a match of the template in the image, deletes the centre pixel. They are iterative algorithms, which erodes the outer layers of pixel until no more layers can be removed ^{[8][9]}. We have used Zhang-Suen methods which use Connectivity numbers to mark and delete pixels. This skeletonization algorithm is a parallel method that means the new value obtained only depend on the previous iteration value. It is fast and simple to be implemented.

Algorithm

This algorithm is made by two sub-iterations. In the first one, a pixel I (i, j) is deleted if the following conditions are satisfied ^{[8][9]}:

1. Its connectivity number is one.
2. It has at least two black neighbours and not more than six.
3. At least one of $I(i,j+1)$, $I(i-1,j)$, and $I(i,j-1)$ are white.
4. At least one of $I(i-1,j)$, $I(i+1,j)$, and $I(i,j-1)$ are white.

In the second sub-iteration the conditions in steps 3 and 4 change:

1. Its connectivity number is one.
2. It has at least two black neighbours and not more than six.
3. At least one of $I(i-1,j)$, $I(i,j+1)$, and $I(i+1,j)$ are white.
4. At least one of $I(i,j+1)$, $I(i+1,j)$, and $I(i,j-1)$ are white.

At the end, pixels satisfying these conditions will be deleted. If at the end of either sub-iteration there are no pixels to be deleted, then the algorithm stops ^{[8] [9]}.

Illustration

In this algorithm it is assumed that the region point in the image has pixel value '1' and background points have value '0'. The Zhang Suen's method consist of successive passes of two basic steps applied to the contour points of the given region, where a contour point is any pixel with value '1' and having at least One 8-neighbor valued '0'. With reference to the 8-neighborhood definition , the first step flags a contour point p for deletion if the following conditions are satisfied ^[10]:

- (a) $2 \leq N(P1) \leq 6$.
- (b) $S(P1) = 1$.
- (c) $P2 * P4 * P6 = 0$.
- (d) $P4 * P6 * P8 = 0$

Where $N(p1)$ is the number of nonzero neighbours of $p1$ that is,

$N(P1) = P2 + P3 + \dots + P8 + P9$ and $S(p1)$ is the number of 0-1 transition in the ordered sequence of $p2, p3, \dots, p8, p9$ ^[10].

Step 1 is applied to every border pixel in the binary region under consideration. If one or more of the conditions (a) through (d) area violated, the value of the point in question is not changed. If all conditions area satisfied the point is flagged for deletion. It is important to be considered, that the point is not deleted until all border points have been processed ^[10].

In the second step, conditions (a) and (b) remain the same, but conditions (c) and (d) are changed to:

- (c') $P2 * P4 * P8 = 0$
- (d') $P2 * P6 * P8 = 0$

After step 1 has been applied to all border points, those that were flagged are deleted, changed to '0' for example. Then, step 2 is applied to the resulting data in exactly the same manner as step 1 ^[10].

5.5 Character Segmentation

This algorithm is basically based on gap between characters and intra-word connections.

(Some of the idea and contents are taken from the paper "A character segmentation algorithm for off-line handwritten script recognition" by Yung, Lai* and Chua issued in 1995) ^[3].

Goal and concept of the algorithm

The goal of the algorithm is to segment the characters with minimum width in a word with at least 90% of segmentation success rate.

The concept of the algorithm is derived from the ideas as below

- i) The horizontal point that separates two thinned characters are either a clear space or a single pixel. By calculating the X-axis projection, this can be determined easily.
- ii) A stroke connecting two characters must be a monotonic function either increasing or decreasing. This function is characterized by its almost identical X-axis and Y-axis projections. Further it is basically different from the projections of most characters.

Writing styles and constraints

- The first constraint is that words on a page must be properly written and separated by detectable gaps. Although connected words do not impose any problems to the segmentation algorithm, it will introduce problems during word recognition or spell checking.
- The second constraint is that lines should be near horizontal.
- Slightly slanted lines are tolerable
- The third constraint is that lines must not overlap with each other.
- We are considering the characters 'd', 'g', 'y' because of their significant characteristics.

Algorithm

1. A thinned image of a word is taken as input
2. X axis projection is calculated on the thinned image
3. Based on the X axis projection, the columns are marked as,
 GAP, if number of pixels in that column is 0
 SINGLE, if number of pixels in that column is 1
 NO_M, if number of pixels in that column is more than 1
4. Check for SINGLE between two NO_Ms. If none found, go to 6.
5. Mark the SINGLES as NO_M
6. Check for a NO_M between two SINGLES. If none found go to 8
7. Mark the NO_Ms as SINGLE
8. Check for consecutive GAPS, if none found, go to 10
9. Merge the GAPS into one GAP at the mid point
10. Check for consecutive SINGLES, if none found, go to 12
11. Mark their mid-point as MID_PT
12. Consider MID_PT and GAP as segmentation points. For each pair of segmentation, calculate the Y axis projection
13. Based on the Y axis projection, the rows are marked as,
 GAP, if number of pixels in that row is 0
 SINGLE, if number of pixels in that row is 1
 NO_M, if number of pixels in that row is more than 1
14. Within this region, check if the number of NO_M greater than the number if SINGLE, if true, go to 16
15. Within this region, check if the number of SINGLE + no of NO_M is greater than the number of GAPS. If false, go to 17
16. Mark the pair of segmentation points as CHECK
17. Check if there is any MID_PT left, if false, END
18. Determine the number of MID_PT between a pair of GAP or CHECK
19. Check if the number of such MID_PT is less than or equal to 2, if true, mark the MID_PT as NO_M, else mark the MID_PT closest to CHECK or GAP as NO_M
20. END

Drawbacks

This algorithm cannot separate overlapping characters. Also certain characters, like m,n,u,v,w etc., which contains a single connecting segment in the character itself, cannot be separated effectively using the used algorithm.

5.6 Template Matching

We are using template matching to recognize required character from an image dataset. A template is a small image (sub-image). The goal is to find occurrences of this template in a larger image, that is, we want to find matches of this template in the image.

Algorithm

The basic approach of the algorithm is ^[11]:

1. For each Image coordinate i,j
2. For the size of the template s,t
3. Compute a pixel-wise metric between the image and the template
4. Sum – next – record the similarity
5. Next
6. A match is based on the closest similarity measurement at each (i,j)

For similarity criteria we are using correlation coefficient approach ^[11]:

Template matching function using correlation:

1. The normalized correlation response between two images f and t is defined as: -

$$C(x, y) = \frac{\sum_{x,y} [f(x,y) - f'] [t(x,y) - t']}{(\sum_{x,y} [f(x,y) - f']^2 [t(x,y) - t']^2)^{1/2}} \quad (4)$$

2. Pick the C(x,y) with the maximum response.

Drawbacks

The drawbacks of this normalized correlation coefficient-based template matching algorithm are computationally expensive and slower than the other approaches.

6. Implementation Details

6.1 Flowchart 1

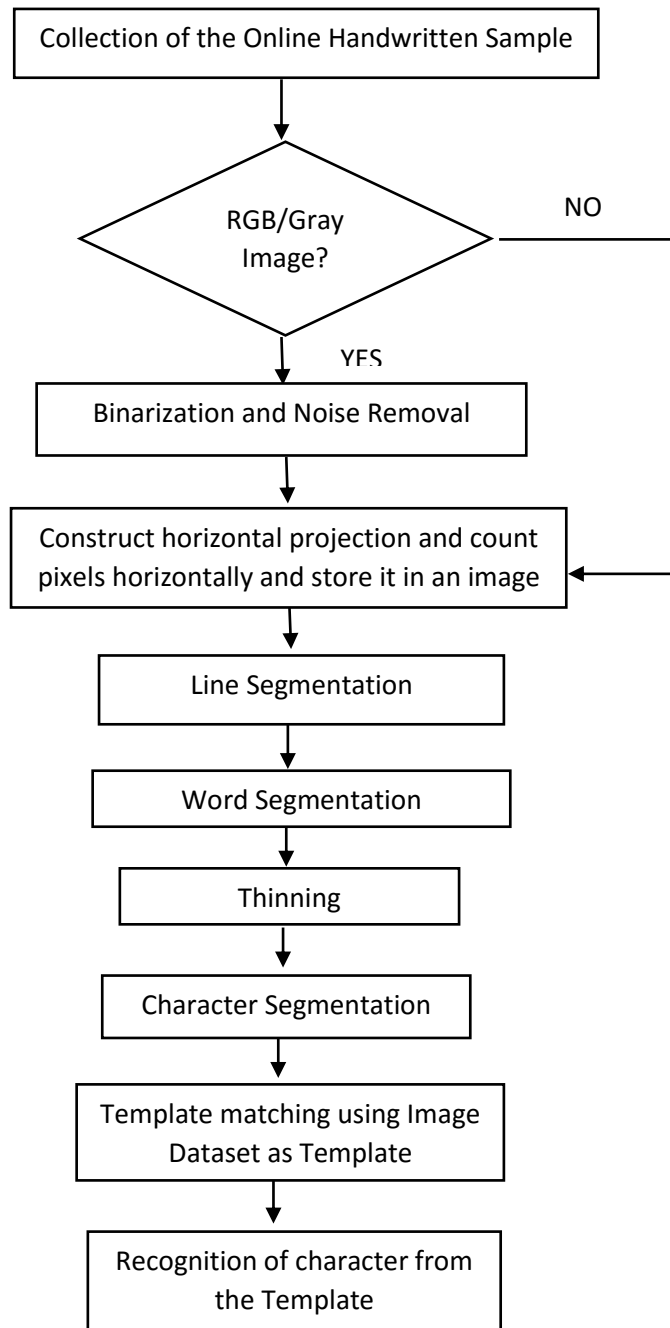
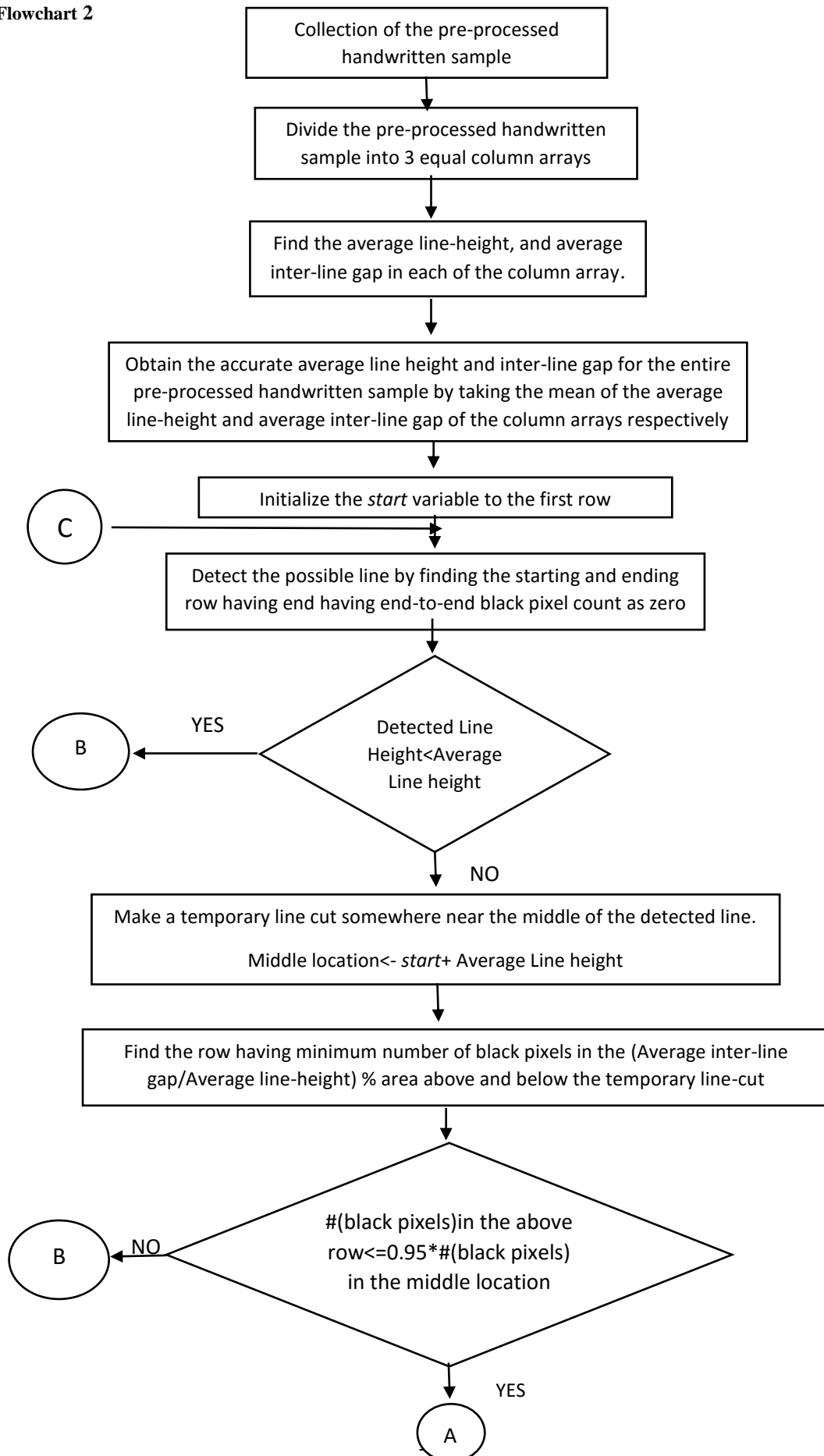


Fig 1: Flowchart for Overall Processing

6.2 Flowchart 2



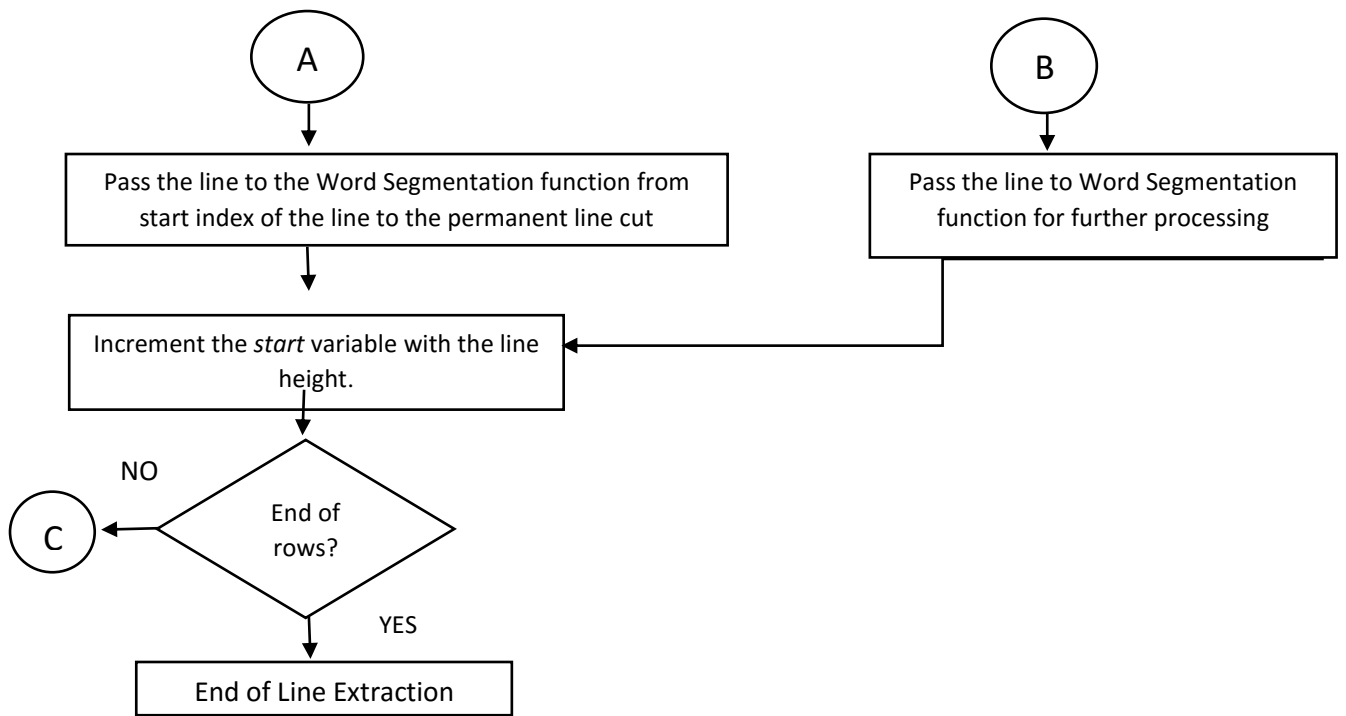


Fig 2: Flowchart for Line Segmentation

6.3 Flowchart 3

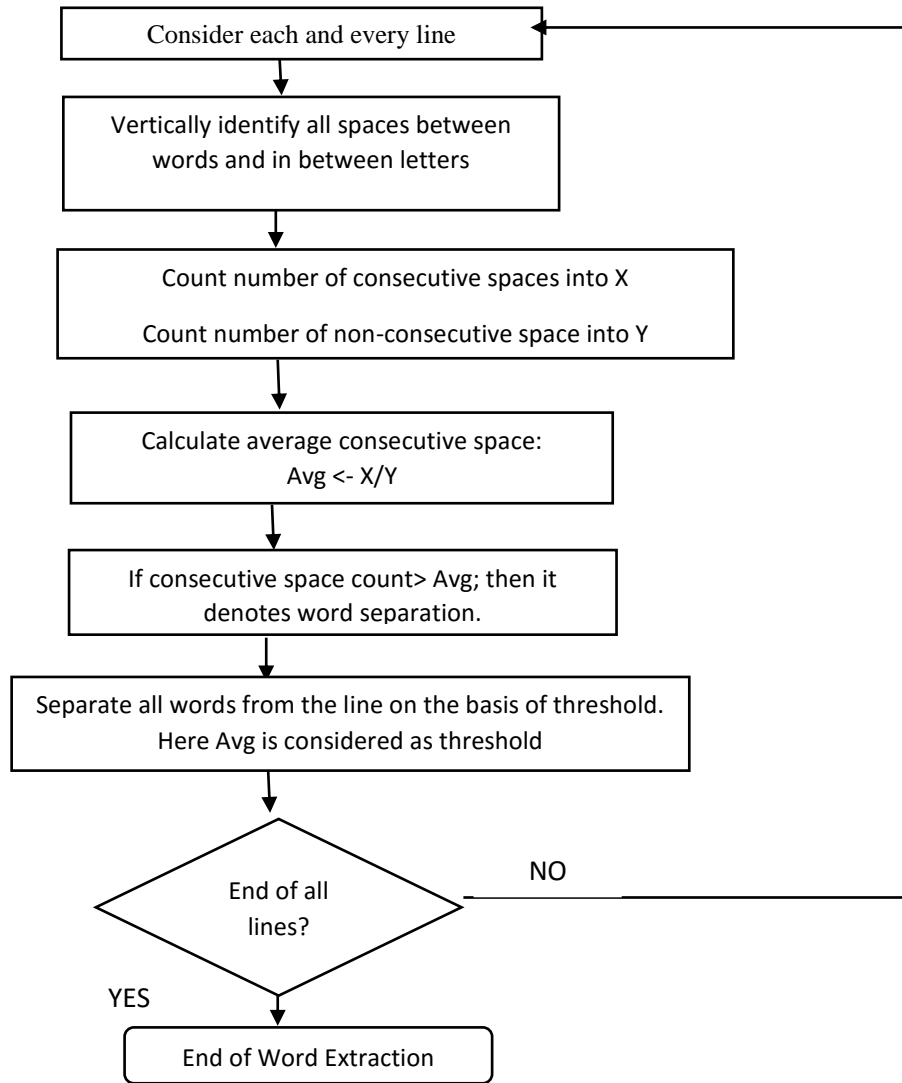
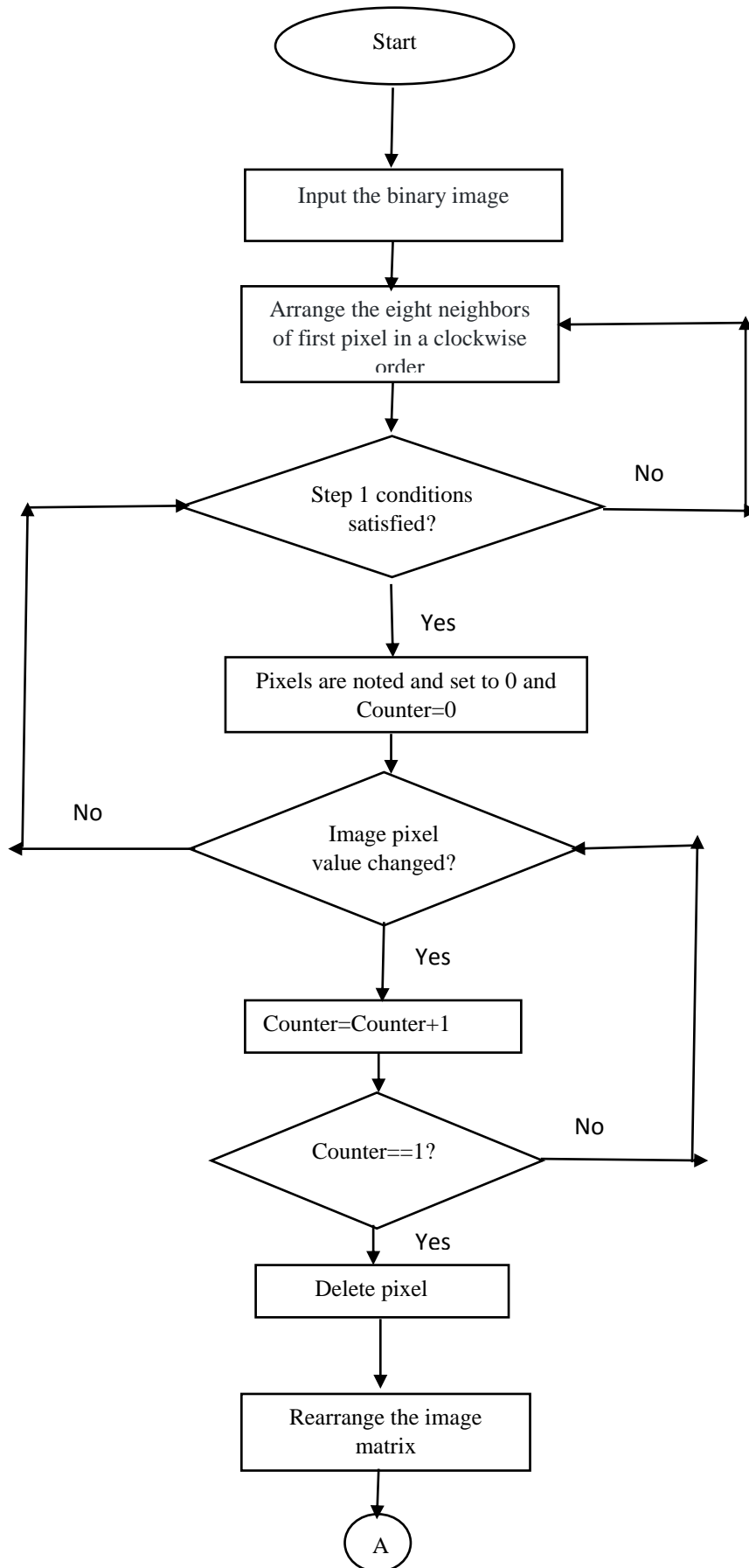


Fig 3. Flowchart for Word Segmentation

6.4 Flowchart 4



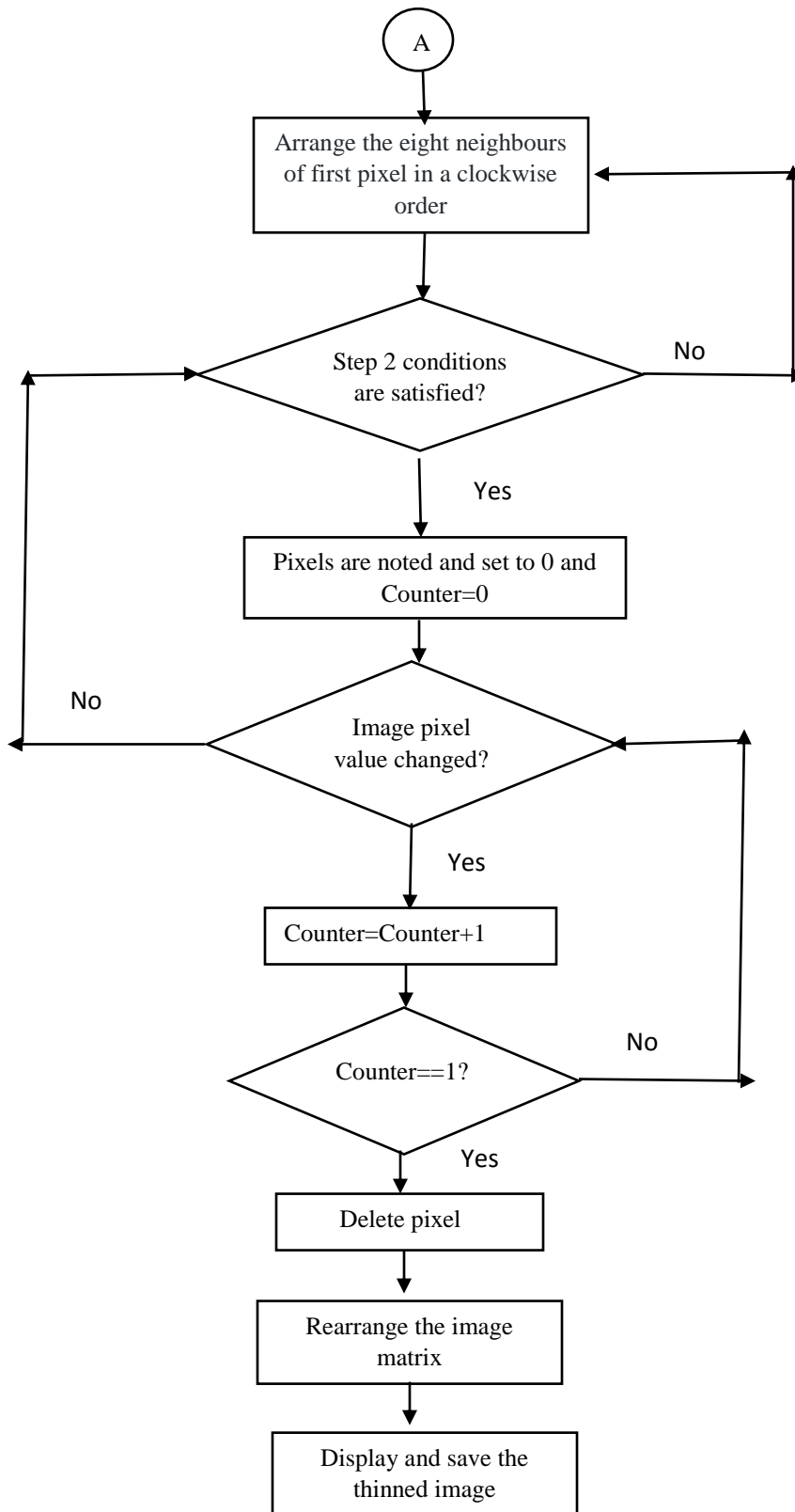
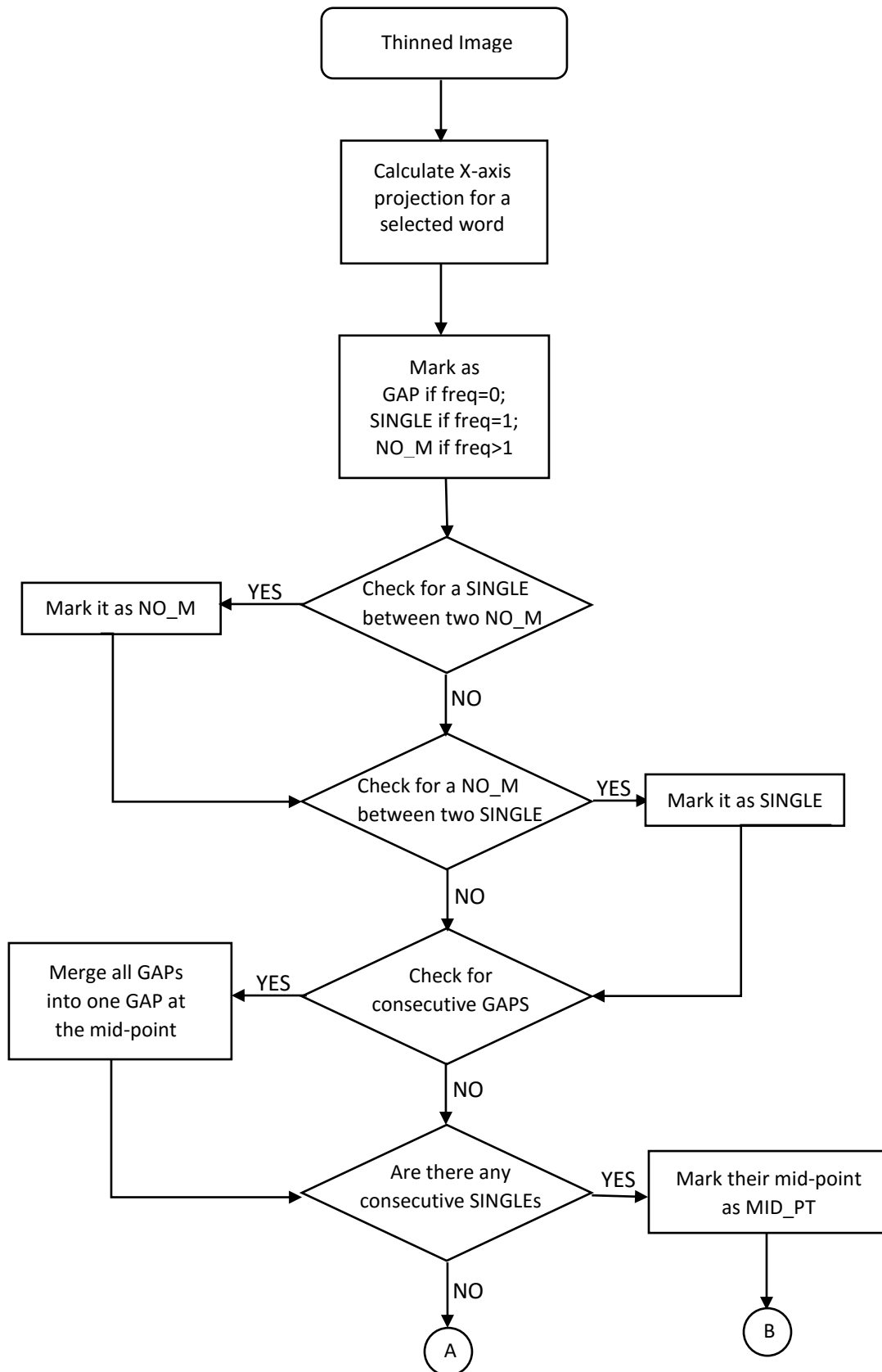


Fig. 4. Flowchart for Thinning

6.5 Flowchart 5



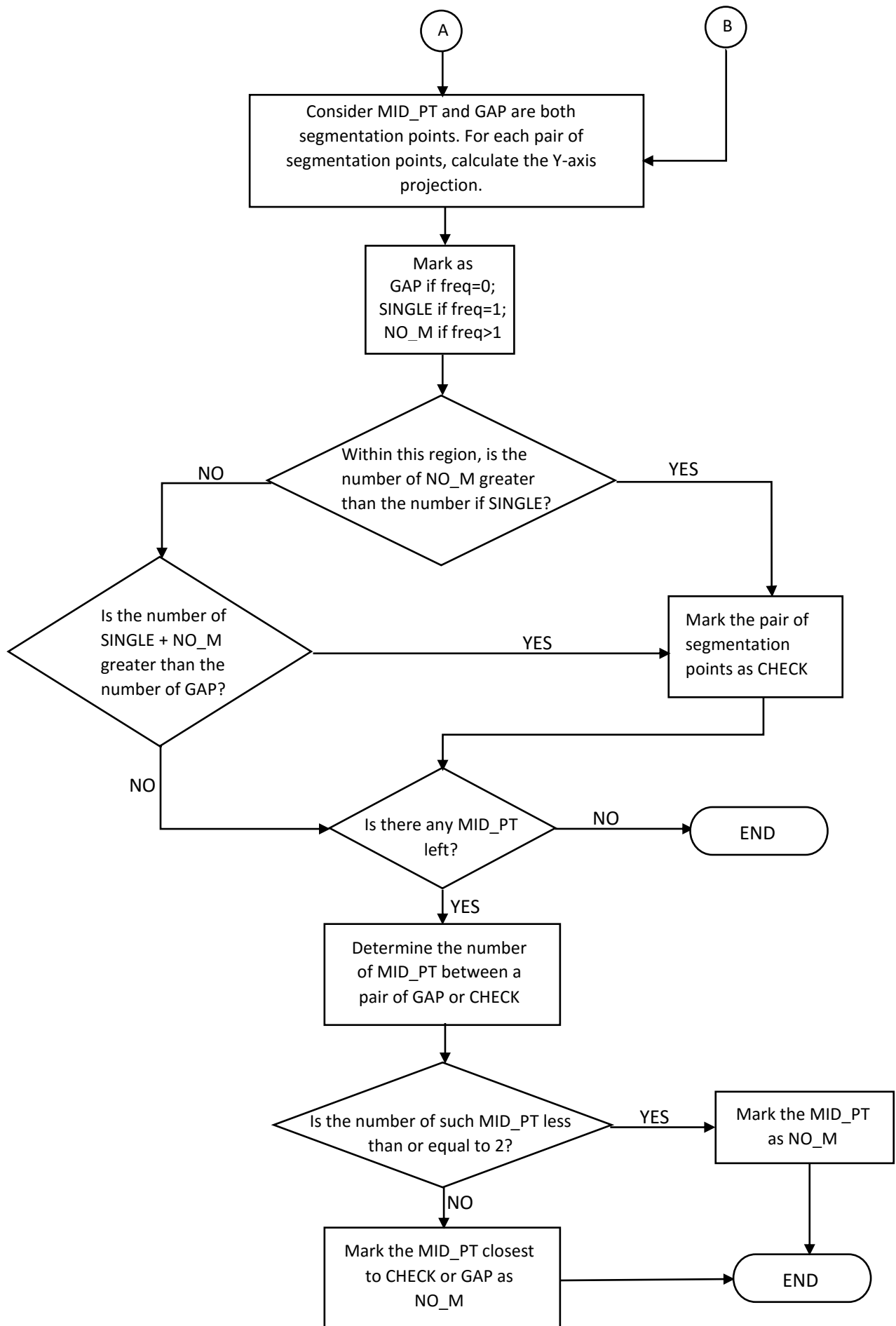


Fig. 5. Flowchart for Character Segmentation

6.6 Flowchart 6

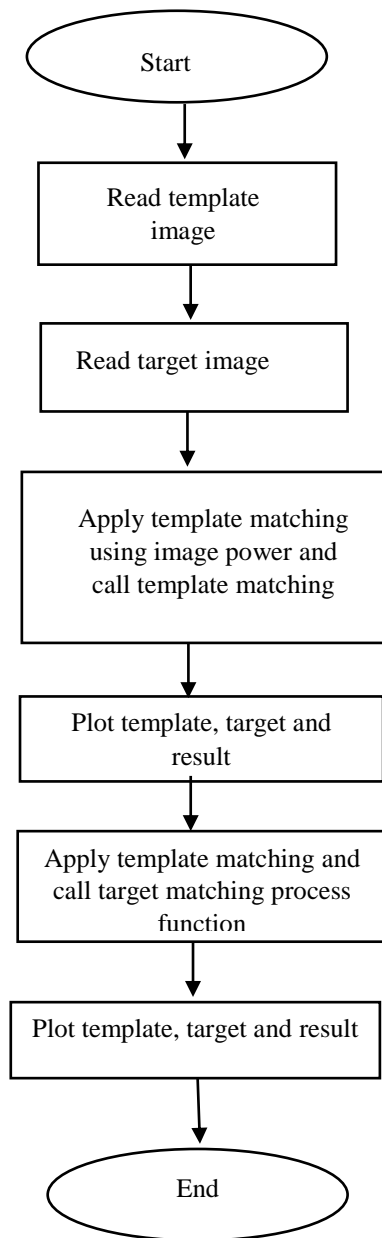


Fig. 6. Flowchart for Template Matching

Flowchart for associated functions

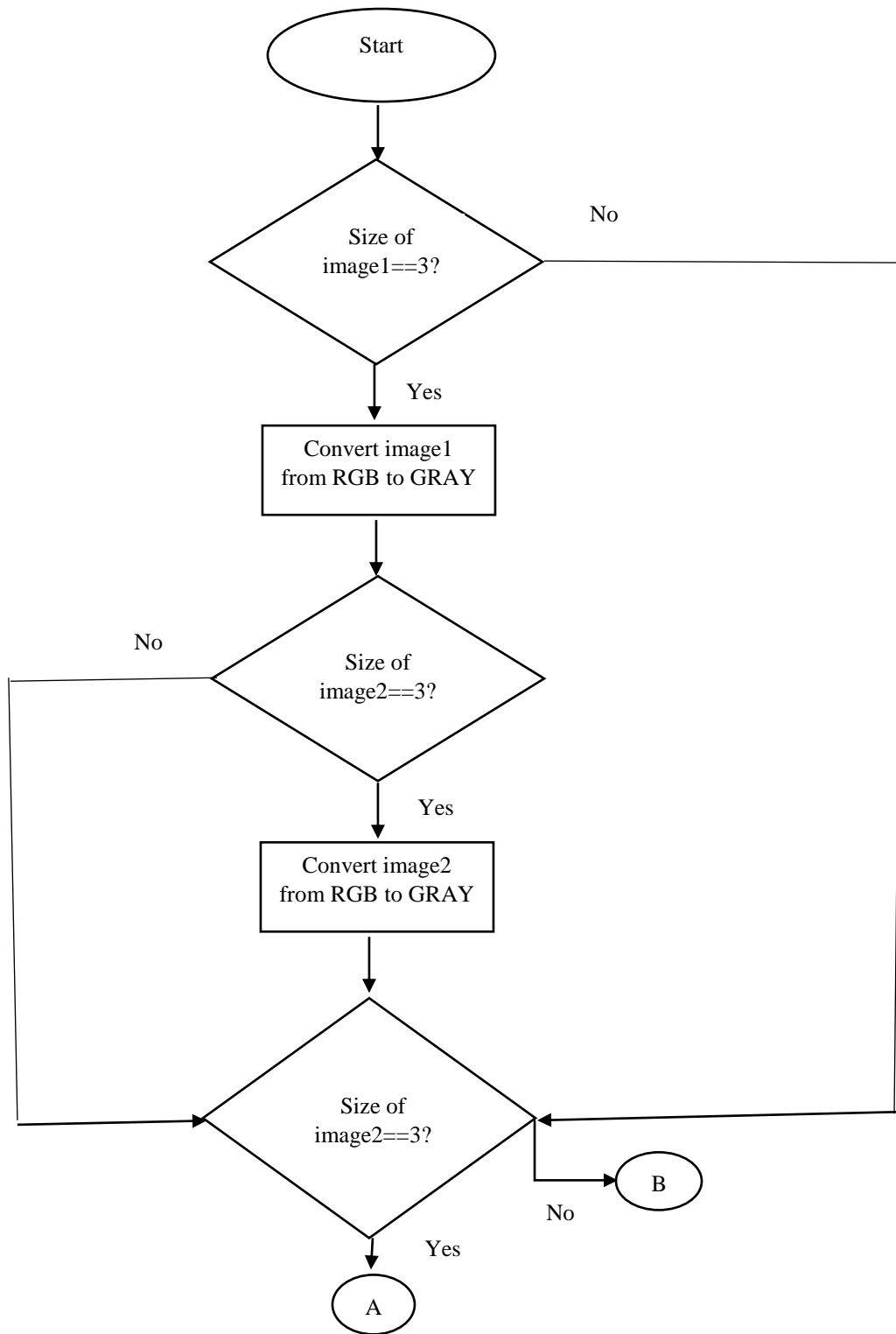


Fig. 7. Flowchart for associated functions for Template Matching

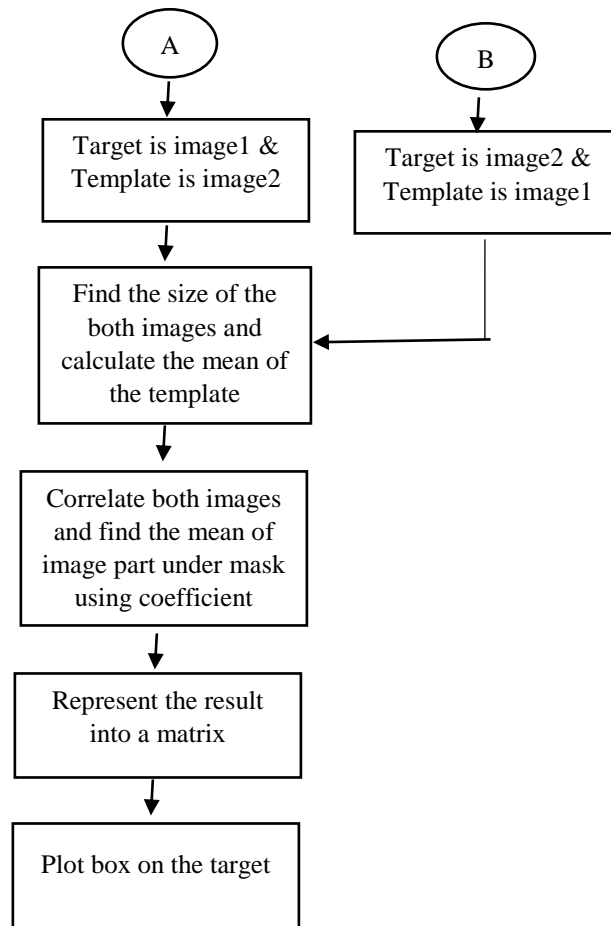
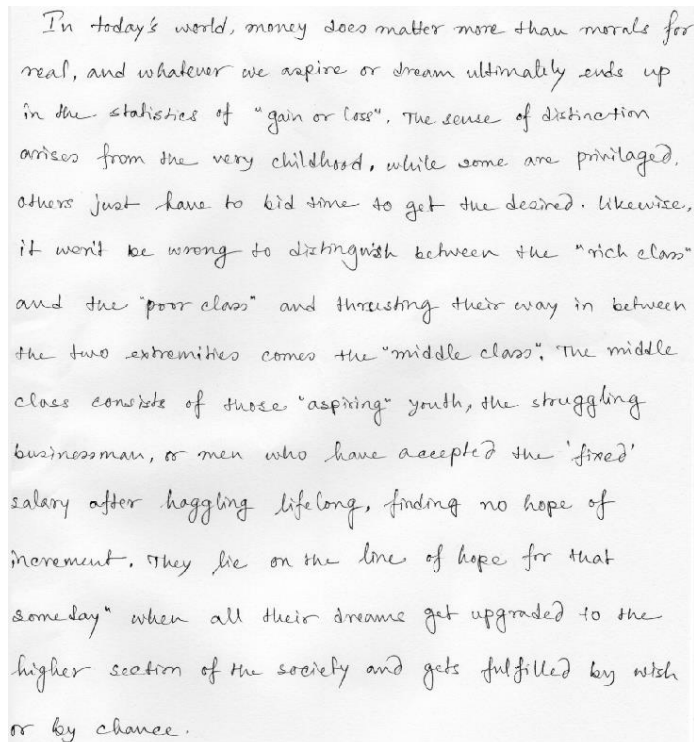


Fig. 8. Flowchart for associated functions for Template Matching (contd.)

7. Results/Sample output

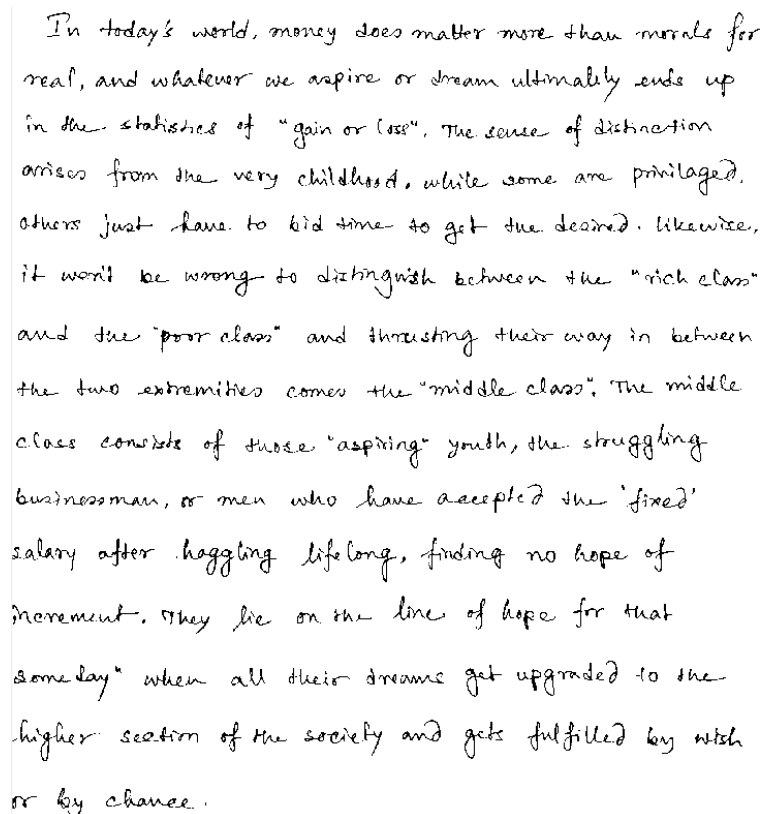
7.1 Pre-processing (converting to grayscale, noise removal, binarization)

- **Original Scanned Document:**



In today's world, money does matter more than morals for real, and whatever we aspire or dream ultimately ends up in the statistics of "gain or loss". The sense of distinction arises from the very childhood, while some are privileged, others just have to bid time to get the desired. Likewise, it won't be wrong to distinguish between the "rich class" and the "poor class" and thrusting their way in between the two extremities comes the "middle class". The middle class consists of those "aspiring" youth, the struggling businessman, or men who have accepted the 'fixed' salary after haggling lifelong, finding no hope of increment. They lie on the line of hope for that "some day" when all their dreams get upgraded to the higher section of the society and gets fulfilled by wish or by chance.

- **Document after pre-processing:**



In today's world, money does matter more than morals for real, and whatever we aspire or dream ultimately ends up in the statistics of "gain or loss". The sense of distinction arises from the very childhood, while some are privileged, others just have to bid time to get the desired. Likewise, it won't be wrong to distinguish between the "rich class" and the "poor class" and thrusting their way in between the two extremities comes the "middle class". The middle class consists of those "aspiring" youth, the struggling businessman, or men who have accepted the 'fixed' salary after haggling lifelong, finding no hope of increment. They lie on the line of hope for that "some day" when all their dreams get upgraded to the higher section of the society and gets fulfilled by wish or by chance.

Fig. 9. Pre-processing (converting to grayscale, noise removal, binarization)

7.2 Line Segmentation:

- **Output: Segmented Lines of the above pre-processed document:**

1. In today's world, money does matter more than morals for
2. real, and whatever we aspire or dream ultimately ends up
3. in the statistics of "gain or loss". The sense of distinction
4. arises from the very childhood, while some are privileged,
5. others just have to bid time to get the desired. Likewise,
6. it won't be wrong to distinguish between the "rich class"
7. and the "poor class" and thrusting their way in between
8. the two extremities comes the "middle class". The middle
9. class consists of those "aspiring" youth, the struggling
10. businessman, or men who have accepted the 'fixed'
11. salary after haggling lifelong, finding no hope of
12. increment. They live on the line of hope for that
13. "someday" when all their dreams get upgraded to the
14. higher section of the society and gets fulfilled by wish
15. or by chance.

Fig.10. Segmented Lines of the above pre-processed document

7.3 Word Segmentation:

- **Input:** (Line No:09 from the Line Segmentation Output):

class consists of those "aspiring" youth, the struggling

Output: Segmented Words:

class

consists

of

those

"aspiring"

youth,

the

struggling

Fig.11. Segmented Words of Line no. 9

- Input: (Line No:13 from the Line Segmentation Output):

some day" when all their dreams get upgraded to the

Output: Segmented Words

some day"

when

all

their

dreams

get

upgraded

to

the

Fig.12. Segmented Words of Line no. 13

7.4 Thinning Output:

i)

Word:

"aspiring"

Thinned Image:

"aspiring"

ii)

Word:

youth,

Thinned Image:

youth,

iii)

Word:

dreams

Thinned Image:

dreams

Fig.12. Thinned Images

7.5 Character Segmentation:

i)

Word:

aspiring

Segmented Word:

aspiring

ii)

Word:

youth

Segmented Word:

youth

iii)

Word:

dreams

Segmented Word:

dreams

Fig.13. Segmented Characters

7.6 Template Matching

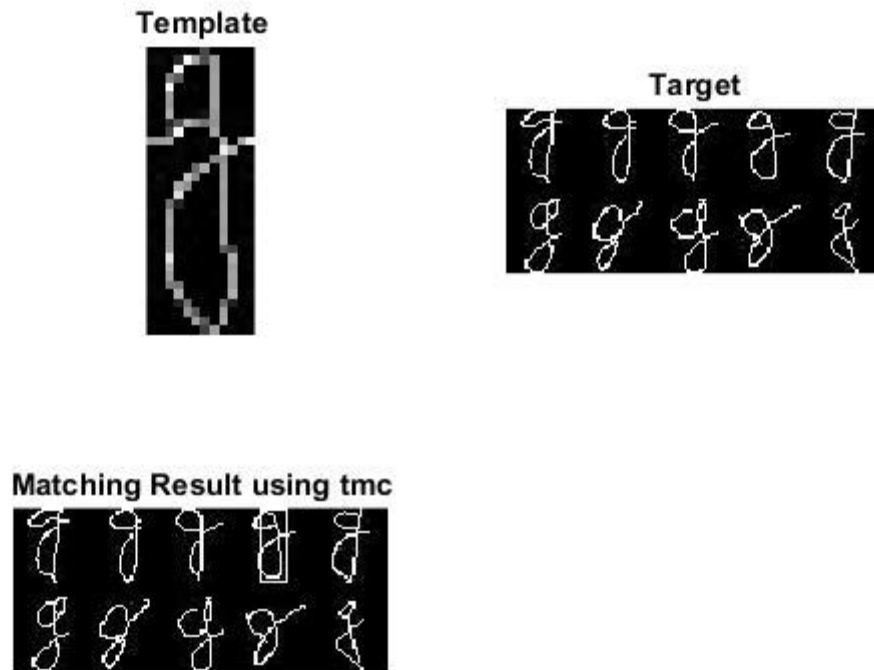


Fig.14. Template Matching for the character 'g'

Predicted Characteristics:

- 1) Large, full loop: good imagination, dramatic ability, friendly, interested in physical activity, "on display", strong libido.
- 2) Long downstroke: many varied interests, restless, perhaps narrow minded.

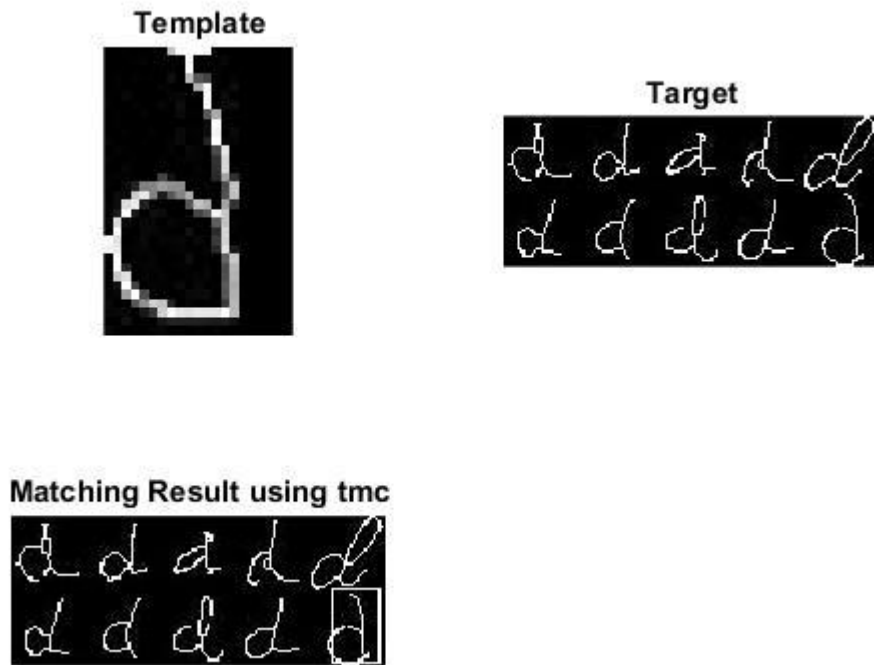


Fig.15 Template Matching for the character 'd'

Predicted Characteristics:

- 1) Tall and narrow: Idealistic and/or religious aspect present.
- 2) Closed oval: Secretive about individual affairs.

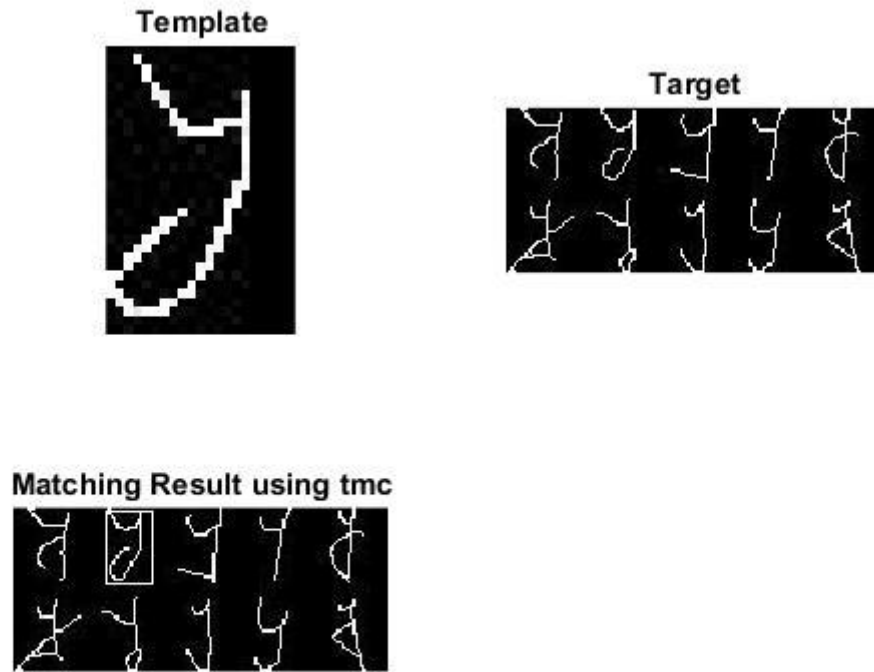


Fig.16. Template Matching for the character 'y'

Predicted Characteristics:

- 1) Open loop: unknowledgeable or unfulfilled concerning sex and/or money.

8. Conclusion

Handwriting analysis is an emerging field for personality recognition. The various personality analysis techniques of this field can be implemented for getting correct personality trait information. Although graphology is an established science, due to human error and ambiguity in the handwriting sample, accuracy of handwriting analysis is found to be around 90% correct prediction. Automated personality identification through handwriting analysis will prove to be a good and helpful system for personality traits identification. It can find its applications especially in the fields of personnel recruitment, in fields of marketing, medicine and counselling and also biometrics, forensics studies (analyzing handwriting on ransom notes in kidnapping cases or blackmailing letters or in the cases of pen poison letters).

References

1. Abhishek Bal and Rajib Saha: An improved Method for Handwritten Document Analysis using Segmentation, Baseline Recognition and Writing Pressure Detection. In: 6th International Conference on Advances In computing & Communications, ICACC 2016, 6-8 September 2016, Cochin, India
2. Parmeet Kaur Grewal, Deepak Prashar: Behavior Prediction Through Handwriting Analysis. IJCST Vol. 3, Issue 2, April - June 2012. ISSN: 0976-8491 (Online) ISSN : 2229-4333 (Print)
3. Nelson H C Yung, Andrew H S Lai* and Perry ZP Chua: A character segmentation algorithm for off-line handwritten script recognition. In: Visual communications and image processing, Taipei, Taiwan, China, 24-26 May 1995, v. 2501, p. 1656-1667
4. C.C. Tappert, C.Y. Suen, T. Wakahara: ON-LINE HANDWRITING RECOGNITION - A SURVEY. CH2614-6/88/0000/1123\$01.00 □ 1988 IEEE
5. R. Plamondon, S.N. Srihari: Online and off-line handwriting recognition: a comprehensive survey. In: IEEE Transactions on Pattern Analysis and Machine Intelligence (Volume: 22, Issue: 1, Jan 2000) INSPEC Accession Number: 6525227
6. Thesis submitted by Rohit Mittal under the supervision of Mr. Khushneet Jindal: Detection and Segmentation of Text in Handwritten Hindi Documents submitted by Computer Science and Engineering Department, Thapar University, Patiala. June 2015
7. Pritam Dhande (Department of Computer Engineering , Pimpri Chinchwad College of Engineering, Pune, India) , Reena Kharat (Department of Computer Engineering , Pimpri Chinchwad College of Engineering, Pune, India): Recognition Of Cursive English Handwritten Characters. International Conference on Trends in Electronics and Informatics ICEI 2017
8. <http://www.ppgia.pucpr.br/~facon/Afinamento/Thinning%20Algorithm.doc>
9. IMAGE PROCESSING TECHNIQUES FOR MACHINE VISION
Alberto Martin and Sabri Tosunoglu
Florida International University
Department of Mechanical Engineering
10. Implementation of an Image Thinning Algorithm using Verilog and MATLAB
Ashwini S. Karne^{a*} and S. S. Navalgund^a Department of Electronics and Communication Engineering, SDMCET, Dharwad, India
International Journal of Current Engineering and Technology
ISSN 2277 – 4106 ©2013 INPRESSCO.
11. <http://www.vis.uky.edu/~ryang/Teaching/cs635-2016spring/Lectures/17-recognition.pdf>

Appendix

Code 1 (Line Segmentation)

```
function Line_Segmentation(filename)
b = imread(filename); %Reading the filename passed to the function
[rows, columns, numberOfColorChannels] = size(b); %gets the no. of rows, columns and
%no. of color channels in the passed filename
if numberOfColorChannels > 1
    img = rgb2gray(b); %Conversion of an RGB image to grayscale
else
    img=b;
end
figure,imshow(img);
img2 = medfilt2(img); %Median Filter applied
img1 = imbinarize(img2); %Binarization of the image
figure,imshow(img1);
%Dividing the image into six equal column arrays
id = fix(columns/6.0);
c1 = img1(:,1:id);
c2 = img1(:,id+1:2*id);
c3 = img1(:,2*id+1:3*id);
c4 = img1(:,3*id+1:4*id);
c5 = img1(:,4*id+1:5*id);
c6 = img1(:,5*id+1:6*id);
%Calling function Count_Lines to determine the average height,average inter-line gap of the lines
%obtained in each column array
[avg_height1,space_height1] = Count_Lines(c1);
[avg_height2,space_height2] = Count_Lines(c2);
[avg_height3,space_height3] = Count_Lines(c3);
[avg_height4,space_height4] = Count_Lines(c4);
[avg_height5,space_height5] = Count_Lines(c5);
[avg_height6,space_height6] = Count_Lines(c6);
%Calculating the average line height and average inter-line gap of all the
%columns combined
avg_height_line= fix((avg_height1+ avg_height2+ avg_height3+ avg_height4+ avg_height5+
avg_height6)/6.0);
avg_inter_line_gap=fix((space_height1+space_height2+space_height3+space_height4+space_height5+space_
height6)/6.0);
%Counting the sum of black pixels in each row of the entire image
sum=zeros(rows,1);
for i= 1:rows
    p=0;
    for j= 1:columns
        if (img1(i,j)== 0)
            p=p+1;
        end
    end
    sum(i)=p;
end
%Line Extraction process
start = 1;
flag = 0;
i=1;
while i<=rows
    if(sum(i)==0 && flag == 0)
        start=i;
        i=i+1;
    elseif(sum(i)==0 && flag == 1)
        stop=i;
        flag=0;
    end
end
```



```

height = stop-start;
if(height<= avg_height_line)
    figure,imshow(img1(start:stop,:));
end
if(height>avg_height_line)
    middle_location = start + avg_height_line;
    k=avg_inter_line_gap/avg_height_line;
    area_above = middle_location-fix(k*avg_height_line);
    area_below = middle_location+fix(k*avg_height_line);
    row_min_black_pixels = middle_location;
    for j= area_above:area_below
        if(j>rows)
            break;
        end
        if(sum(row_min_black_pixels)>= sum(j) && j~=middle_location)
            row_min_black_pixels =j;
        end
    end
    if(sum(row_min_black_pixels)<=fix(0.95*sum(middle_location)))
        stop=row_min_black_pixels;
        DEMO_LINE=img1(start:stop,:);
        Trimming(DEMO_LINE);
    else
        DEMO_LINE=img1(start:stop,:);
        Trimming(DEMO_LINE);
    end
    start=stop+1;
    i=stop+1;
end
else
    flag=1;
    i=i+1;
end
end
end %end of function Line_Segmentation
%Counting the number of lines
function [average_height,space_height]= Count_Lines(column_part)
[rows, columns] = size(column_part);
sum=zeros(rows,1);
for i= 1:rows
    p=0;
    for j= 1:columns
        if (column_part(i,j)== 0)
            p=p+1;
        end
    end
    sum(i)=p;
end
start=1;
flag = 0;
no_of_lines=0;
space_height=0;
height = zeros(rows,1);
for i= 1:rows
    if(sum(i)==0 && flag == 0)
        start=i;
        space_height=space_height+1;
    elseif(sum(i)==0 && flag == 1)
        stop=i;
        flag=0;
        no_of_lines = no_of_lines+1;
    end
end

```

```

        height(no_of_lines)= stop-start;
    else
        flag=1;
    end
end
end
total_height=0;
for i=1:no_of_lines
    total_height = total_height + height(i);
end
average_height = fix(total_height/no_of_lines);
space_height=fix(space_height/no_of_lines+1);
end
%function to remove extra white spaces on both sides(if any) of the
%segmented lines
function Trimming(DEMO_LINE)
[rows, columns] = size(DEMO_LINE);
sum=zeros(1,columns);
for i= 1:columns
    p=0;
    for j= 1:rows
        if (DEMO_LINE(j,i)== 0)
            p=p+1;
        end
    end
    sum(:,i)=p;
end
start = 1;
i=1;
while i<=columns
    if(sum(:,i)==0)
        start=i;
        i=i+1;
    else
        break;
    end
end
stop=columns;
i=columns;
while i>=1
    if(sum(:,i)==0)
        stop=i;
        i=i-1;
    else
        break;
    end
end
figure,imshow(DEMO_LINE(:,start:stop));
LINE=DEMO_LINE(:,start:stop);
Word_Segmentation_Demo(LINE);
end

```

Code 2 (Word Segmentation)

```
function Word_Segmentation_Demo(LINE)
[rows, columns] = size(LINE); %gets the no. of rows, columns
img1 = medfilt2(LINE); % Median Filter applied
%Calculating the sum of black pixels in each column of the segmented line
sum=zeros(columns,1);
for i= 1:columns
    p=0;
    for j= 1:rows
        if (img1(j,i)== 0)
            p=p+1;
        end
    end
    sum(i)=p;
end
gapsw=zeros(columns,3);
index=1;
flag=0;
%Measuring width of each intra-word and inter-word and then storing the
% width into the array gapsw[]
for i= 1:columns
    if (sum(i)==0 && flag==0)
        if(sum(i+1)==0)
            start=i;
            flag=1;
        elseif(sum(i+1)~=0)
            gapsw(index,1)=1; %single gap detected
            gapsw(index,2)=i;
            gapsw(index,3)=i;
            index=index+1;
        end
    elseif(sum(i)==0 && flag==1 && sum(i+1)~=0)
        stop=i+1;
        % disp(stop)
        flag=0;
        gapsw(index,1)=stop-start; %consecutive gap detected
        gapsw(index,2)=start;
        gapsw(index,3)=stop;
        index=index+1;
    end
end
tgp=0;
twd=0;
for i=1:columns
    if(gapsw(i,1)~=0)
        tgp=tgp+1;
        twd=twd+gapsw(i,1);
    end
end
threshold=fix(twd/tgp);
start=1;
for i=1:tgp
    if(gapsw(i,1)>=threshold)
        figure,imshow(img1(:,start:(gapsw(i,2)-1)));
        start=gapsw(i,3);
    end
end
end
```

Code 3:(Thinning)

```
%Code for thinning
Img_Original = imread('F:\FINAL YEAR PROJECT\Sample\w2.jpg');
Img_Original = rgb2gray(Img_Original);
% Convert gray images to binary images using Otsu's method
Otsu_Threshold = graythresh(Img_Original);
BW_Original = not(im2bw(Img_Original,Otsu_Threshold)); % must set object region as 1, background region
as 0

changing = 1;
[rows, columns] = size(BW_Original);
BW_Thinned = BW_Original;
BW_Del = ones(rows, columns);
while changing
    % BW_Del = ones(rows, columns);
    changing = 0;
    % Step 1
    for i=2:rows-1
        for j = 2:columns-1
            P = [BW_Thinned(i,j) BW_Thinned(i-1,j) BW_Thinned(i-1,j+1) BW_Thinned(i,j+1)
BW_Thinned(i+1,j+1) BW_Thinned(i+1,j) BW_Thinned(i+1,j-1) BW_Thinned(i,j-1) BW_Thinned(i-1,j-1)
BW_Thinned(i-1,j)]; % P1, P2, P3, ... , P8, P9, P2
            if (BW_Thinned(i,j) == 1 && sum(P(2:end-1))<=6 && sum(P(2:end-1)) >=2 && P(2)*P(4)*P(6)==0
&& P(4)*P(6)*P(8)==0) % conditions
                % No. of 0,1 patterns (transitions from 0 to 1) in the ordered sequence
                A = 0;
                for k = 2:size(P,2)-1
                    if P(k) == 0 && P(k+1)==1
                        A = A+1;
                    end
                end
                if (A==1)
                    BW_Del(i,j)=0;
                    changing = 1;
                end
            end
        end
    end
    BW_Thinned = BW_Thinned.*BW_Del; % the deletion must after all the pixels have been visited
    % Step 2
    for i=2:rows-1
        for j = 2:columns-1
            P = [BW_Thinned(i,j) BW_Thinned(i-1,j) BW_Thinned(i-1,j+1) BW_Thinned(i,j+1)
BW_Thinned(i+1,j+1) BW_Thinned(i+1,j) BW_Thinned(i+1,j-1) BW_Thinned(i,j-1) BW_Thinned(i-1,j-1)
BW_Thinned(i-1,j)];
            if (BW_Thinned(i,j) == 1 && sum(P(2:end-1))<=6 && sum(P(2:end-1)) >=2 && P(2)*P(4)*P(8)==0
&& P(2)*P(6)*P(8)==0) % conditions
                A = 0;
                for k = 2:size(P,2)-1
                    if P(k) == 0 && P(k+1)==1
                        A = A+1;
                    end
                end
                if (A==1)
                    BW_Del(i,j)=0;
                    changing = 1;
                end
            end
        end
    end
end
```

```

    end
    BW_Thinned = BW_Thinned.*BW_Del;
end% while

figure
%subplot(1,2,1)
%imshow(BW_Original)
%subplot(1,2,2)
BW_Thinned=imcomplement(BW_Thinned);
imshow(BW_Thinned);
imwrite(BW_Thinned,'F:\FINAL YEAR PROJECT\Sample\thin2.jpg')

```

Code 4 (Character Segmentation)

```

%code for character segmentation
IM = imread('E:\Project Files\Temp Match\thinW1.jpg');
level = graythresh(IM);
BIN = imbinarize(IM,level);
figure,imshow(BIN)
[r,c] = size(BIN);
k=1;
VP(1:c)=0;
for i=1:c
    count=0;
    for j=1:r
        if(BIN(j,i)==0)
            count=count+1;
        end
    end
    VP(k)=count;
    k=k+1;
end
Mark(1:c)='';
for i=1:c
    if(VP(i)==0)
        Mark(i)='GAP';
    elseif(VP(i)==1)
        Mark(i)='SINGLE';
    else
        Mark(i)='NO_M';
    end
end
end
%Single between two NO_M
for i=2:c-1
    if(Mark(i)=='SINGLE'&&Mark(i-1)=='NO_M'&&Mark(i+1)=='NO_M')
        Mark(i)='NO_M';
    end
end
end
%NO_M between two single
for i=2:c-1
    if(Mark(i)=='NO_M'&&Mark(i-1)=='SINGLE'&&Mark(i+1)=='SINGLE')
        Mark(i)='SINGLE';
    end
end
end
i=1;
%Merge consicutive GAPS at mid point
while i<=c
    if(Mark(i)=='GAP')
        count=0;
        j=i;
        while(Mark(j)=='GAP'&&j<c)

```

```

        Mark(j)='DEL';
        count=count+1;
        j=j+1;
    end
    Mark(i+floor(count/2))='GAP';
    i=j;
end
i=i+1;

end
i=1;
%Mark consicutive SINGLE at mid point
while i<=c
    if(Mark(i)=='SINGLE')
        count=0;
        j=i;
        while(Mark(j)=='SINGLE'&&j<c)
            count=count+1;
            j=j+1;
        end
        Mark(i+floor(count/2))='MID_PT';
        i=j;
    end
    i=i+1;
end
%Horizontal projection
k=1;
HP(1:r)=0;
for i=1:r
    count=0;
    for j=1:c
        if(BIN(i,j)==0)
            count=count+1;
        end
    end
    HP(k)=count;
    k=k+1;
end
%Recalculating HP for segmentation regions
i=1;
while i<c
    if(Mark(i)=='GAP' || Mark(i)=='MID_PT')
        j=i+1;
        %while((Mark(j)=='GAP' || Mark(j)=='MID_PT')&&j<c)
        while((Mark(j)~='GAP' && Mark(j)~='MID_PT')&&j<c)
            j=j+1;
        end
        k=1;
        %Horizointal projection for a region(- highest number of pixels)
        HPr(1:r)=0;
        for i1=1:r
            count=0;
            for j1=i:j
                if(BIN(i1,j1)==0)
                    count=count+1;
                end
            end
            HPr(k)=count;
            k=k+1;
        end
        Markr(1:r)='';
    end
end

```

```

for i1=1:r
    if(HPr(i1)==0)
        Markr(i1)='GAP';
    elseif(HPr(i1)==1)
        Markr(i1)='SINGLE';
    else
        Markr(i1)='NO_M';
    end
end
%counting number of NO_M and SINGLE
c_NO_M=0;
c_SINGLE=0;
c_GAP=0;
for i1=1:r
    if(Markr(i1)=='GAP')
        c_NO_M=c_NO_M+1;
    elseif(Markr(i1)=='SINGLE')
        c_SINGLE=c_SINGLE+1;
    elseif(Markr(i1)=='SINGLE')
        c_GAP=c_GAP+1;
    end
end
if(c_NO_M>c_SINGLE)
    Mark(i)='CHECK';
    Mark(j)='CHECK';
else
    if(c_NO_M+c_SINGLE>c_GAP)
        Mark(i)='CHECK';
        Mark(j)='CHECK';
    end
end
%-----
i=j;
else
    i=i+1;
end
end
i=1;
f=0;
while i<c
    if(Mark(i)=='MID_PT')
        f=1;
        break;
    end
    i=i+1;
end
if(f==1)
    i=1;
    while(i<c)
        if(Mark(i)=='GAP' || Mark(i)=='CHECK')
            j=i+1;
            % while((Mark(j)=='GAP' || Mark(j)=='CHECK')&&j<c)
            while((Mark(j)~='GAP' && Mark(j)~='CHECK')&&j<c)
                j=j+1;
            end
            count=0;
            for i1=i:j
                if(Mark(i1)=='MID_PT')
                    count=count+1;
                end
            end
        end
    end
end

```

```

        if(count<=2)
            for i1=i:j
                if(Mark(i1)=='MID_PT')
                    Mark(i1)='NO_M';
                end
            end
        else
            for i1=i:j
                if(Mark(i1)=='MID_PT')
                    Mark(i1)='NO_M';
                    break;
                end
            end
            for i1=j:-1:i
                if(Mark(i1)=='MID_PT')
                    Mark(i1)='NO_M';
                    break;
                end
            end
        end
        i=j;
    end
    i=i+1;
end
end
i=1;
k=1;
folder='E:\Project Files\Temp Match';
while(i<c)
    j=i+1;
    while((Mark(j)~='GAP' && Mark(j)~='CHECK')&&j<c)
        j=j+1;
    end
    CHAR=BIN(:,i:j);
    figure,imshow(CHAR);
    imwrite(CHAR,fullfile(folder,sprintf('c%d.jpg',k)));
    k=k+1;
    i=j+1;
end

```

Code 5 (Template Matching)

```

% code for template matching
% main file

```

```

close all
clear all

```

```

% read Template image
im=imread('t1.jpg');
im1=imcomplement(im);
%im1=imread('S.bmp');
%im1=imread('image1.jpg');

```

```

% read Traget Image
im2=imread('temp.jpg');
%im2=imread('image2.jpg');

```

```

% apply templete matching using power of the image
result1=tmp(im1,im2);

```

```

figure,

```



```

subplot(2,2,1),imshow(im1);title('Template');
subplot(2,2,2),imshow(im2);title('Target');
subplot(2,2,3),imshow(result1);title('Matching Result using tmp');

% apply template matching using DC components of the image
result2=tmc(im1,im2);

figure,
subplot(2,2,1),imshow(im1);title('Template');
subplot(2,2,2),imshow(im2);title('Target');
subplot(2,2,3),imshow(result2);title('Matching Result using tmc');

%associated function :tmc(image1,image2)

function result=tmc(image1,image2)
if size(image1,3)==3
    image1=rgb2gray(image1);
end
if size(image2,3)==3
    image2=rgb2gray(image2);
end
% check which one is target and which one is template
if size(image1)>size(image2)
    Target=image1;
    Template=image2;
else
    Target=image2;
    Template=image1;
end
% read both images sizes
[r1,c1]=size(Target);
[r2,c2]=size(Template);
% mean of the template
image22=Template-mean(mean(Template));
% Target=double(Target);
% Template=double(Template);
%corrolate both images
corrMat=[];
for i=1:(r1-r2+1)
    for j=1:(c1-c2+1)
        Nimage=Target(i:i+r2-1,j:j+c2-1);
        Nimage=Nimage-mean(mean(Nimage)); % mean of image part under mask
        corr=sum(sum(Nimage.*image22));
        corrMat(i,j)=corr;
    end
end
% plot box on the target image
result=plotbox(Target,Template,corrMat);

%associated function :tmp(image1,image2)

function result=tmp(image1,image2);
if size(image1,3)==3
    image1=rgb2gray(image1);
end
if size(image2,3)==3

```

```

    image2=rgb2gray(image2);
end
% check which one is target and which one is template using their size
if size(image1)>size(image2)
    Target=image1;
    Template=image2;
else
    Target=image2;
    Template=image1;
end
% find both images sizes
[r1,c1]=size(Target);
[r2,c2]=size(Template);
% mean of the template
image22=Template-mean(mean(Template));
% correlate both images
M=[];
for i=1:(r1-r2+1)
    for j=1:(c1-c2+1)
        Nimage=Target(i:i+r2-1,j:j+c2-1);
        Nimage=Nimage-mean(mean(Nimage)); % mean of image part under mask
        corr=sum(sum(Nimage.*image22));
        warning off
        M(i,j)=corr/sqrt(sum(sum(Nimage.^2)));
    end
end
% plot box on the target image
result=plotbox(Target,Template,M);

```

%associated function : plotbox(Target,Template,M)

```

function result=plotbox(Target,Template,M);
[r1,c1]=size(Target);
[r2,c2]=size(Template);
[r,c]=max(M);
[r3,c3]=max(max(M));
i=c(c3);
j=c3;
result=Target;
for x=i:i+r2-1
    for y=j
        result(x,y)=255;
    end
end
for x=i:i+r2-1
    for y=j+c2-1
        result(x,y)=255;
    end
end
for x=i
    for y=j+c2-1
        result(x,y)=255;
    end
end
for x=i+r2-1
    for y=j+c2-1
        result(x,y)=255;
    end
end
end

```