Universal Graphical User Interface for Online Handwritten Character Recognition

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

Master of Engineering
in
Computer Science and Engineering

Submitted By
Mayur Vyas
(801232014)

Under the supervision of:
Karun Verma
Assistant Professor

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004

June 2014

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, **"*Universal Graphical User Interface for Online Handwritten Character Recognition*"**, in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Computer Science and Engineering* 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. Karun Verma* 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.

Mayur Vyas

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

Mr. Karun Verma
Assistant Professor,
Computer Science and Engineering Department

Countersigned by

(Dr. Deepak Garg)
Head
Computer Science and Engineering Department
Thapar University
Patiala

(Dr. S. K. Mohapatra)
Dean (Academic Affairs)
Thapar University
Patiala

Acknowledgement

First of all, I would like to thank my family members who are dearest and precious to me for their love, encouragement, blessings and support in all respects. Most importantly, none of this would have been possible without the love and patience of my family. My immediate family, to whom this dissertation is dedicated to, has been a constant source of love, concern, support and strength all these years.

I am extremely thankful to my respective guide ***Mr. Karun Verma*** of Thapar University for his valuable guidance, advice, motivation, encouragement, moral support, sincere effort and positive attitude with which he solved my queries and provide delightful ambiance for learning, exploring and making this thesis possible. It has been a great pleaser and experience to work under his sanctuary.

Dr. Deepak Garg is one of the best teachers that I have had in my life. He sets high standards for his students and he encourages and guides them to meet those standards.

I would like to thank my colleague and wife ***Shivangi*** for her continuous support and help by which research work involved in the thesis has been progressed towards completion.

Abstract

Handwritten character segmentation is the process of dividing a character into multiple segments so that it can be passed to the recognition system and on the basis of these segments the recognition system can recognize that particular character. In this process user provides the input through any touch devices like PDA, Mobiles or by the Tablets and the segmentation tool takes the input and break the character into segments. There are various tools available which performs the segmentation but no one is available in all the platforms.

The aim of this thesis is to represent a methodology which does not segment the characters on the basis of their shapes. Instead of segmenting the character on the basis of their shapes, it divides the character into the points. It finds the co-ordinate of the every point where pen move from the pen down to pen up. This is a platform independent tool that runs online on any browser which is HTML5 enabled. Since it does not depend upon the shape of the character, it segments more efficiently than other tools.

Table of Content

List of Figures

Chapter 1
Introduction

1.1 Handwriting Character Recognition

Handwriting character recognition is the ability of a computer to take and understand handwritten character as input from sources such as paper documents, mobiles, touch-pads, photographs, and any other devices. Several types of interpretation, analysis and recognition may be associated with handwritten character. Handwriting character recognition is a process of transforming a language represented character into its symbolic representation. A handwriting recognition system should handle formatting of the character and should perform correct segmentation of the characters to find the most possible words.

For the devices with a stylus or digital pen, such as that used with a Mobile phone, Personal Digital Assistant (PDA) devices and Tablet PC, the handwritten recognition technique is important to take characters and symbols as input. A classifier, which does not depend on writer, is constructed using them and training examples acquired from many individuals. However, the recognition performance is insufficient, since the size, style and shape of handwritten patterns varies among several users. Generally a specific user uses the device which is for personal use. If user properly corrects the recognition errors at the time of writing then the class label input patterns can be acquired. Therefore, a classifier is constructed which is dependent on the writer. It's considered that the user-specific classifier is better in the terms of performance than that of the generic one. However, for obtaining a high performance user specific classifier, a large number of training examples should be gathered. This requires lot of time and effort to be spent for collecting the samples. To solve this problem, several methods have been proposed. In this method, for constructing a classifier which understands the input patterns of a specific user, a prepared generic classifier is updated or modified every time the user writes any pattern. As a result, it has been reported that even when very few samples per

class have been entered the recognition error rate can be reduced by the constructed classifier [1].

Most of the handwritten character recognition system uses linguistic knowledge to improve the recognition accuracy of the system. For the tasks which use limited vocabulary, such as language models based on lexicon, address reading are used successfully. On the other side, character level (character n-grams) language models can bring advantage to a task with the large vocabulary.

Handwriting is a digitized, dynamic representation of the pen movement, which describes sequential information about acceleration, position, velocity, or pen angles as a function of time. Every component contains the sequence of pen tip information, which is sampled from the movement of writer's pen. Here the horizontal and vertical coordinates are referred. When the pen tip touches the surface pen down components are recorded and when the pen is lifted pen up components are recorded.

In handwritten character recognition, mostly the character recognition is done on the basis of the stroke identification. A character is divided into multiple parts which are called strokes and then these strokes are matched to the stroke database. By matching the strokes from the database that stroke is get identified and by merging all the strokes of that particular character system may also identify the character. For the handwriting recognition stroke-level Hidden Markov Model [2] and segmental Hidden Markov Model [3] may be used.

The handwritten character recognition procedure includes data collection and their analysis, segmentation, feature extraction, training, classification, stroke identification from stroke database and then recognition of the character. Data is capture through the user by the pen and then it pass to the preprocessing step. Segmentation and feature extraction is the pre-activities for the system as shown in the figure 1.1.

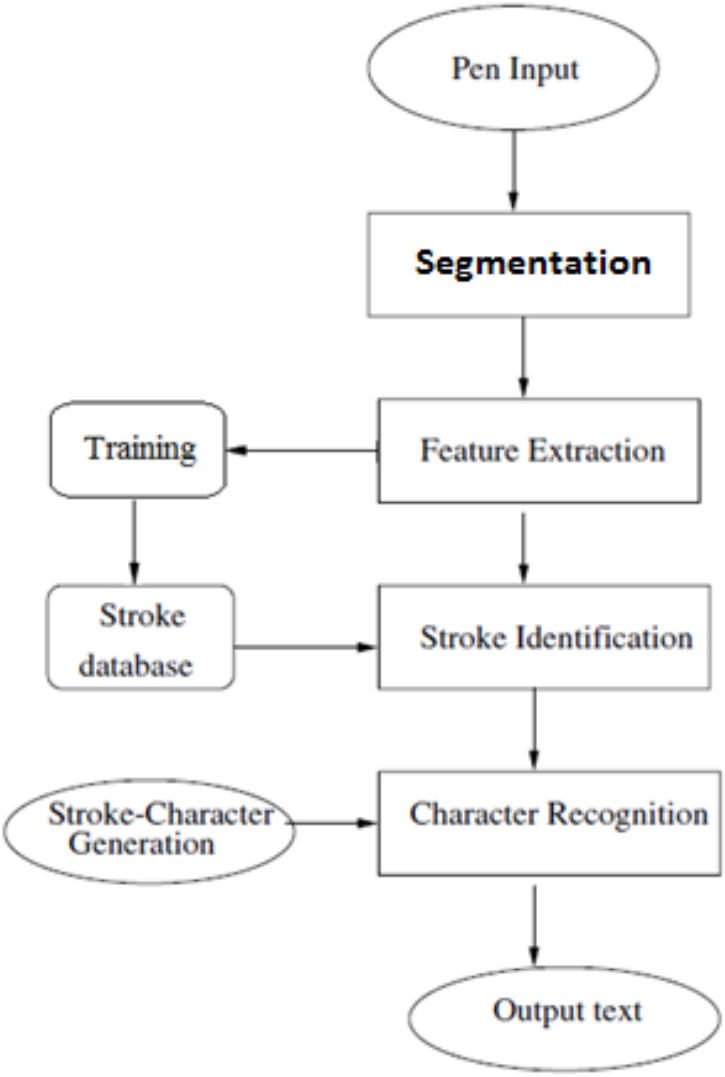


Fig.1.1 Model Design of Handwritten character recognition system

As shown in the figure 1.1, first of all system takes the input from the any tough sensitive devise through any compatible pen, stylus or finger tips. After tacking the input, that handwritten input is further segmented into the multiple parts. These segmented parts are called the strokes. These strokes are used for the feature extraction and they these features are used for the training of the system. After training of the system, these strokes are stored into the stroke database where they have assigned a unique stroke ID. When a new stroke is occurs then that stroke is matched to the stroke which is stored into the stroke database. After matching the stroke that stroke may be identified. By generating a

character from these identified strokes, system can identify the character which is given by the user as handwritten input.

In this proposed approach, any handwritten character is a combination of various strokes and every stroke is measured according to the movement of the pen tip on the writing area (from pen down to pen up). When pen tip makes the contact with the writing surface then a new stroke generated and when the pen tip leaves the writing surface then this stroke is completed. Therefore the very first step for the handwritten character segmentation is to find the various strokes from the given character. For finding the strokes of the characters we divide the character in multiple parts and this process is called segmentation. When the stroke is identified we can easily identify the appropriate character.

In our approach, the handwritten character is represented as a combination of various strokes. A stroke is defined as the movement by the pen tip from the instant when it makes contact with the writing surface to the earliest moment when the contact with the writing surface is broken. Therefore, the very first step in the handwritten character recognition is to recognize all the component strokes in a character and we can do this by the segmentation. Once the segmentation is done and the strokes are identified, the character itself can be easily identified.

1.2 Handwritten Character Segmentation

Online handwritten character segmentation is essential process in which sequence of continuous handwritten characters are decomposed into sub-strokes for the recognition system. In this process, the feature of an unknown stroke is founded so that it can be compared with a database of such strings. This stroke database contains the feature strings of all the strokes that form handwritten Characters. There may be multiple variations of a single stroke. All the variations of any single stroke are given a common identity which is called the stroke ID. Then the stroke identification consists of mapping an unknown string of stroke features onto a stroke ID. As shown in the figure 1.2 segmentation is a process of understanding and generation. Segmentation system understands the handwritten input and then divide the input characters into multiple parts

which is called the generation. Handwritten input is passed to the segmentation tool and then this tool segments the handwritten input.

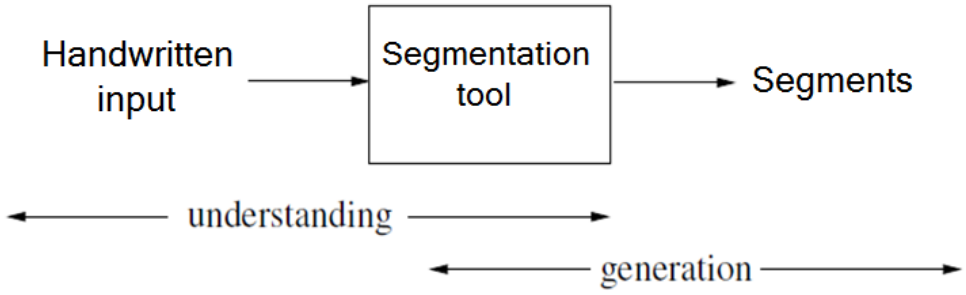


Figure 1.2 Block Diagram of Segmentation

Handwritten character segmentation is the process of dividing a character into multiple segments so that it can be passed to the recognition system and on the basis of these segments the recognition system can recognize that particular character. In this process user provides the input through any touch devices like PDA, Mobiles or by the Tablets and the segmentation tool takes the input and break the character into segments. There are various techniques available which performs the segmentation of the handwritten input. The aim of this paper is to do survey of various techniques and find out the best among them.

Segmentation of handwritten character is one of the most important processes to do for the better handwriting recognition. If a character is not properly segmented then it may cause to false recognition of the character. The segmentation of the handwritten characters is a difficult task because there are many writing styles for a single character and characters may be written in a way to touch each other or to overlap with each other. As far as the segmentation is concerned, the character writings in any language may be classified into the following types.

- Handwritten input characters should not be connected or overlapped. They should be separated from each other and should be isolated.
- The components by which that handwritten characters is composed should also not be overlapped. They should be written too far apart.

- Adjacent characters are written closely in such a way that the bounding boxes of these characters over- lap with each other but these characters don't touch each other.
- Adjacent characters are written closely in such a way that they touch each other but the bounding boxes of these characters do not overlap.
- Adjacent characters are written closely in such a way that they touch each other and also their bounding boxes overlap.

1.3 On-line and Off-line Recognition

In online handwriting recognition, the system takes the input on the touch device and when user writes the system also performs the recognition process on the back end. When user pen ups then it is completion of a stroke and after the completion of a stroke system tries to identify that stroke. By identifying the strokes and merging them, system identifies the handwritten character which is given as input.

In offline handwriting recognition, system takes the input from the file which is written previously in past and then system tries to recognize the character of that input file. Offline handwriting recognition system generally takes an image as input and then tries to find out the boundary within the characters are written and then it tries to segment them. After segmenting these characters, the offline handwriting recognition system tries to identify these strokes and the characters.

In the on-line handwriting recognition, system automatic converts the text which is written through special digitizer or mobile or any PDA devices, where a sensor picks the pen-tip movements as well as it also pick the points where pen-up and pen-down so that system can easily recognize the pen-up/pen-down switching. This type of data is called as digital ink and can be treated as a digital representation of handwritten character. The captured signals are converted into the letter codes which can be used within text-processing applications and computer. Figure 1.3 shows typical input signals that can be analyzed in both cases.

In off-line character recognition, document that is written in past is scanned by the system. This means the every individual character will need to be extracted which is contained in the scanned image. There are many tools available that are capable of performing this task. However, there are many common imperfections in this task. The most common task is when the characters that are connected to each other are returned as a single sub-image which contains both characters. It creates a major problem in the stage when characters are recognized. However, there are many algorithms are available which reduce the risk of connected characters.

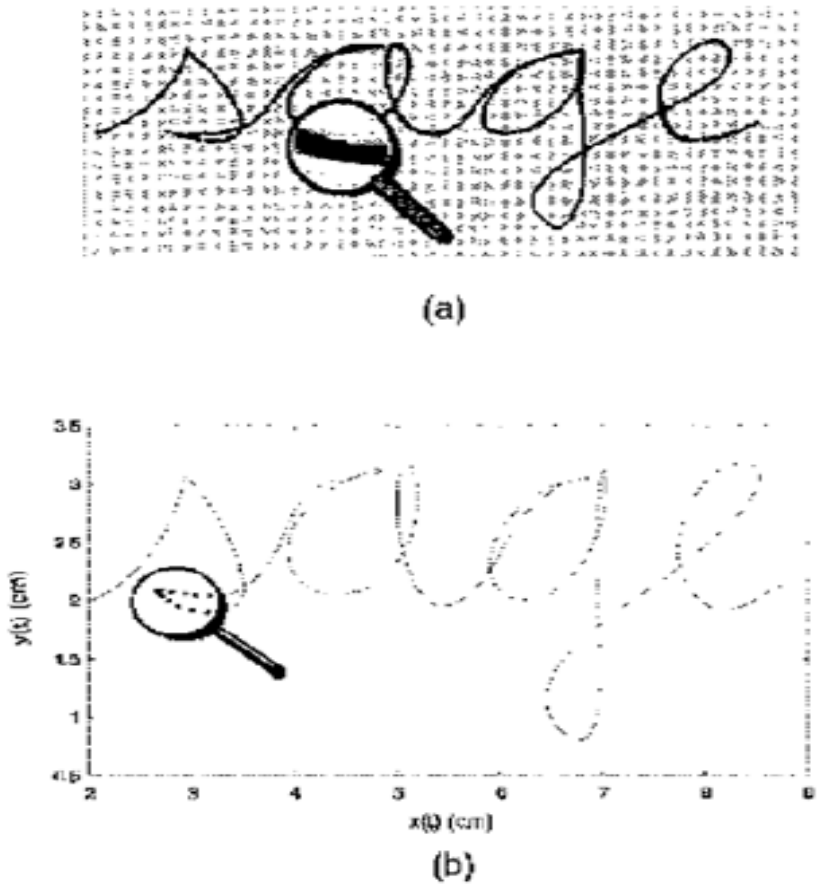


Figure 1.3 (a) Off-line handwritten character segmentation,(b) On-line handwritten character segmentation [4].

The raw data storage requirements in off-line and on-line handwritten character recognition are widely different. The data requirements in the on-line handwriting recognition (figure 1.2b) for an average cursively written word are a few hundred bytes

per second and in the case of off-line handwriting recognition (Figure 1.2a), few-hundred kilo-bytes. The recognition rates reported are much higher in case of on-line recognition comparison with the off-line recognition [5].

On-line handwriting recognition means that while the user writes the machine recognizes the writing. Sometimes the term dynamic or real time has been used in place of online. Depending on the speed of the computer and the recognition technique, the recognition lags behind the writing to a lesser or greater extent. On-line handwriting recognition systems need to be fast enough to keep up with the writing. Average writing rate of a human is 1.5-2.5 characters per second for English alphanumeric language. Peak writing rate can approach to 5-10 characters per second, for example a sequence of 1's can be written quickly.

Off-line handwriting recognition is generally performed after user completes the writing. The scanners have x and y resolutions of typically 250-400 points per inch. Optical character recognition (OCR) is a superset of offline handwriting recognition. Generally most of the OCR work has been on machine-printed characters, there has been considerable effort on handwriting as well [6], [7]. Optical character recognition systems typically process hundreds of characters per second.

An advantage of the on-line handwriting recognition devices over the off-line handwriting recognition devices is that they capture the temporal or dynamic information of the writing. This temporal or dynamic information consists of the speed of the writing within each stroke, the direction of the writing for each stroke, the number of strokes and the order of the strokes. Pen movement from pen down to pen up is called a stroke. Many of on-line transducers capture the line drawing as a sequence of coordinate points or trace of the handwriting. On the other hand, off-line conversion of scanner data to line drawings usually requires imperfect preprocessing. On-line entry provides the temporal information which improves recognition accuracy [8]. In some experiments, the on-line recognized data were converted to the form of off-line data to show that on-line is superior to off-line recognition on the same data. Conversely, off-line recognized handwritten data have been converted by line thinning to sequences of points similar to

on-line recognized handwritten data (but without the information of timing), achieving reasonable accuracy of recognition [9].

1.4 Handwritten Input

The movements of the pen tip may be sensed “on line”, for example by a pen-based PDA surface or pen-based computer screen surface. In the on-line handwriting recognition case, the 2-D coordinates of successive points of the writing as a function of time are stored in order, i.e., the order of strokes made by the writer is readily available. In the off-line handwriting recognition case, the completed writing is available as an image. Alternatively, the image of the previously written text may be sensed “off line” from a piece of paper by intelligent word recognition or optical scanning (optical character recognition [10]).

1.5 Elements of the Handwritten character Segmentation Interface

The elements of the on-line handwriting segmentation interface typically include:

- A stylus or pen from which user write the input.
- Any touch sensitive surface, which may be adjacent to or integrated with, an output display for taking input.
- A software application which captures the movements of the pen or stylus from the writing surface and then translate the resulting stroke into digital text.

1.4 Organization of Thesis

This thesis is organized as followed

Chapter 2: Previous work on the handwritten character segmentation is represented in this chapter.

Chapter 3: What are the various problems we have faced in the handwritten character segmentation is represented in this chapter.

Chapter 4: The objective of this research work and the methodologies used to full file these objectives are represented in this chapter.

Chapter 5: This chapter represents implementation and result of the proposed work.

And the last chapter is devoted to the conclusion.

Chapter Summary

Handwritten character recognition is process of taking handwritten character as input and then understanding it. Handwritten character recognition and its uses are discussed in detail in this chapter. For the better handwriting recognition there is need to perform the handwritten character segmentation. As discussed in this chapter handwritten character segmentation is a process of dividing a character into multiple parts and then extracting the feature of that character. After extracting the feature of the handwritten character, the strokes are used for the handwritten character recognition. There are two type of handwriting character recognition that are online and offline handwritten recognition. These types of handwriting recognition are also discussed in this chapter. Also the handwritten input and various elements of the handwritten character segmentation are discussed briefly in this chapter.

Chapter 2
Literature Survey

The handwritten character segmentation is an important preprocesses of the on-line handwritten character recognition and incorrect segmentation will lead to false recognition of the character. There are some techniques which are previously used for the handwritten character segmentation.

2.1 Nebulous Stroke Models

Nebulous stroke model method is a shape based segmentation method. In this method, the characters are segmented on the basis of their shaper. Different-different characters may have the some similar shapes and these common shapes help to recognize these characters [11].

The drawback of these approaches is that the handwritten sample has to be segmented before the recognition, which generally prematurely limits the hypothesis space. Furthermore, the training of any sub-character model can be difficult task, because while training of the letter models are initiated using samples of individual letters that is hard to obtain reliable samples of isolated sub-character patterns.

A letter model is made by the concatenation of many stroke models. A stroke is the any segment of a handwritten script. In this process the only limitations of the system are the number of strokes in a particular letter, and which strokes can be shared. Manual segmentation is not involved in training: the stroke models trained at first on later on whole word samples and isolated letters, but didn’t train on isolated or segmented strokes. Figure 2.1 shows the segmentation of some handwritten samples when a Hidden Markov Model is used to model. "a," "g," and "j" is segmented in respectively six, eight, and five strokes in the samples shown in the figure 2.1, respectively. "a" and "g" share the 4 strokes S1, ,S4, S5, S6; while "g" and "j" has four strokes S18, S19, S20, S1 common; S l stroke is shared by all three samples.

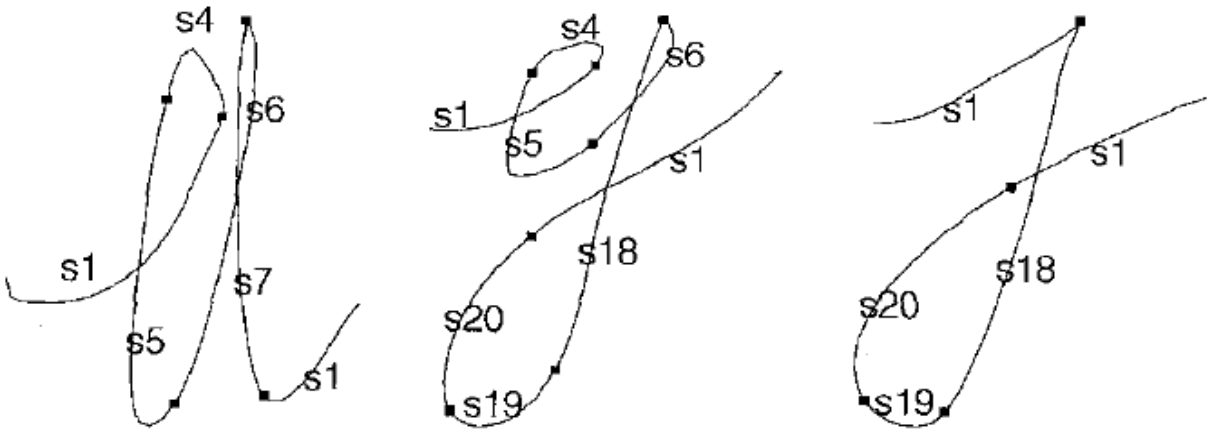


Figure 2.1 Stroke segmentation of three letter samples [12]

2.2 Elastic Matching

There are very large character datasets and because of this there are so many variations in the training data and for overcome this problem we use elastic matching. Elastic matching overcome that problem at some extent [13][14]. Tappert uses the elastic matching technique [15] to match and handwritten cursive character with its possible sequences (figure 2.2).

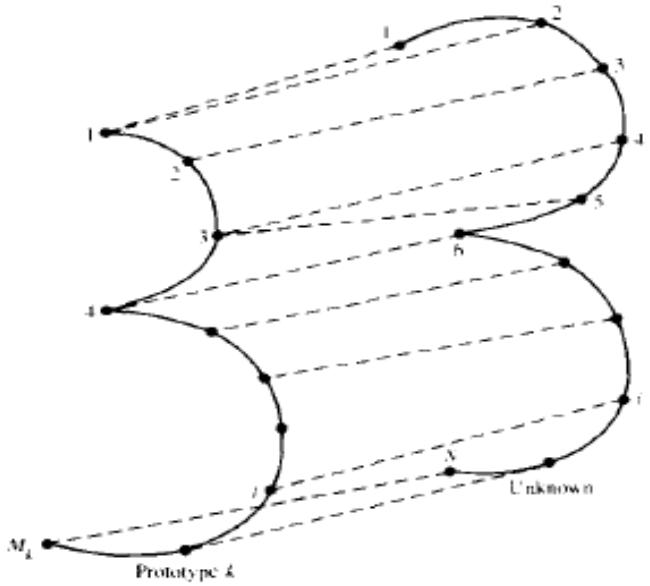


Figure 2.2 Illustration of elastic matching [15]

It is an online method in it the unknown words are represented by means of the angles and y-location of the strokes joining digitization points. In elastic matching the sum of distance between these word feature and the letter prototypes sequences had to be minimized. Segmentation constraints were used to improve the performance and Dynamic programming is used with the warping function to neglect the unnecessary features.

2.3 Analysis-by-synthesis

There is another approach for the character segmentation called analysis-by-synthesis (Figure 2.3), sometimes it also called recognition by generation [16], [17]. This model generally uses strokes and the rules for connecting them to build symbols. The symbols are generated from the inventory of strokes constitute idealized standard representations of symbols. An approximation to the real handwritten character can be attained by specifying strokes with mathematical models that describes the movement of a pen tip as a function of time. Then, a handwritten character can be divided into strokes, then using the model parameters the strokes are classified, and the letter sequences and character recognized [18].

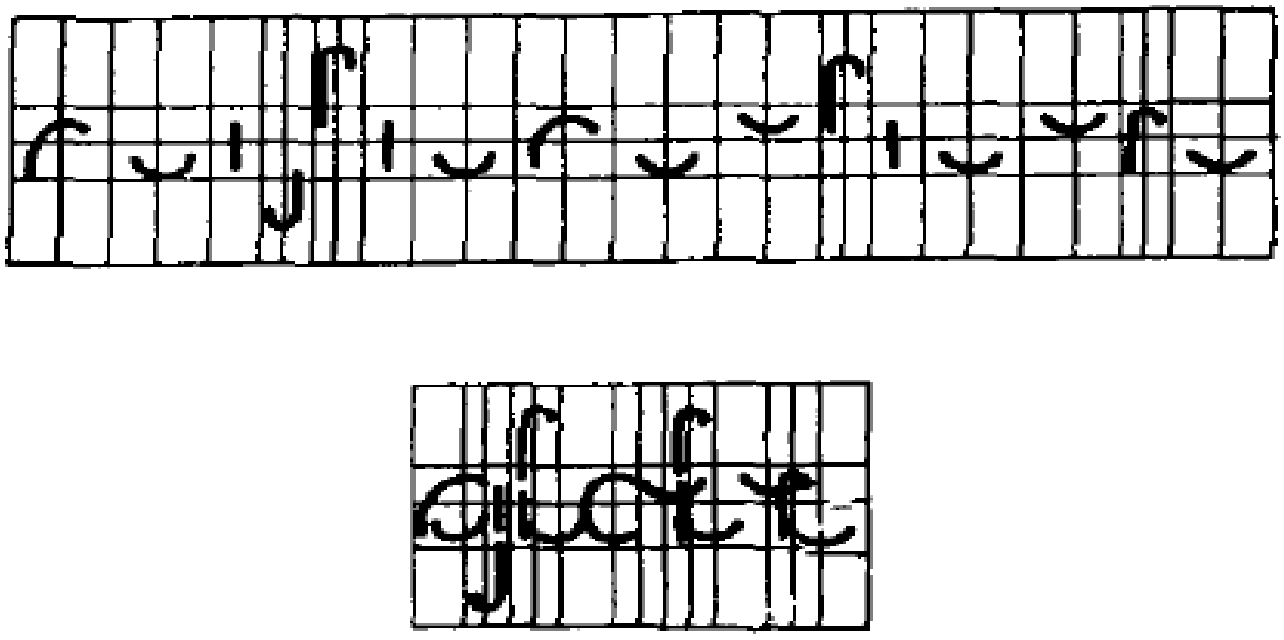


Figure 2.3 Illustration of analysis-by-synthesis [19]

2.4 Dissection Techniques for Segmentation

Dissection techniques are the decomposition of frames into sequence of the sub-frames using the general features (see fig 2.4). Dissection techniques are the process which divides the frames without using any specific information about the shapes of the data.

Figure 2.4 Dissection of the handwritten word

In some systems where classification and segmentation are performed separately and they do not interact with each other, dissection is the entire handwritten character segmentation process. We have saw in many current studies that handwritten character segmentation is a very complex process so there is always a need of dissection to make smaller frames for the segmentation.

2.4.1 White Space and Pitch Removing

In handwritten characters, white spaces are generally used for the separation of the successive words. When we divide the frames into sub-frames then we create separate boxes for the connected characters and do not include the white spaces into these boxes. By applying that rule we can correctly segments the handwritten characters which are overlapped or connected to each other or merged by a line.

User writes the inputs on any particular devise and then an application reads these characters from left to write and removes the white spaces between the characters.

Hoffman and McCullough [20] have generalized that process and in this formulation, the segmentation stage consist the three steps:

(1) Detection of the start of a character.

(2) Sectioning of the character.

(3) Detection of end of the character.

2.4.2 Projection Analysis

The vertical projection is used for counting the black pixels in each column. It is also used for detecting the white space between the successive letters. Figure 2.5 represents the vertical projection of a line which consists of a simple running count of the black pixels in each column.

When the written character touches, or overlaps horizontally, the projection generally contains a minimum at the proper segmentation column. The projection was first obtained, and then the ratio of second derivative of this curve to its height was used as a criterion for choosing separating columns [21].

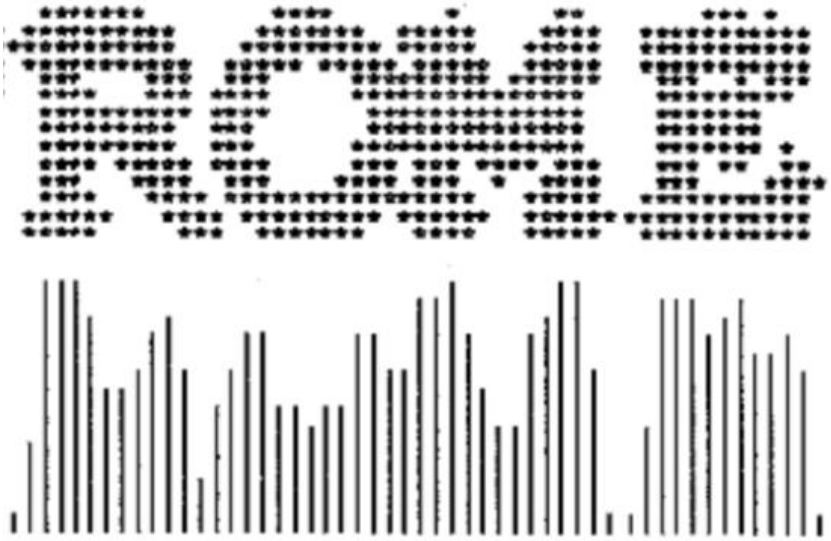


Figure 2.5 Vertical projection of an image [21]

2.4.3 Connected Component Processing

Projection methods are very useful for better character segmentation, where adjacent characters can be separated at columns. A one-dimensional analysis is feasible in such a case.

Segmentation of handwritten character is done by two-dimensional analysis, for even nonteaching characters may not be easily separated along a single straight line. There is a common approach shown in the figure 2.6 which is based on determining connected black regions, the connected components. There some extra processing is necessary to combine or split these components into character images.

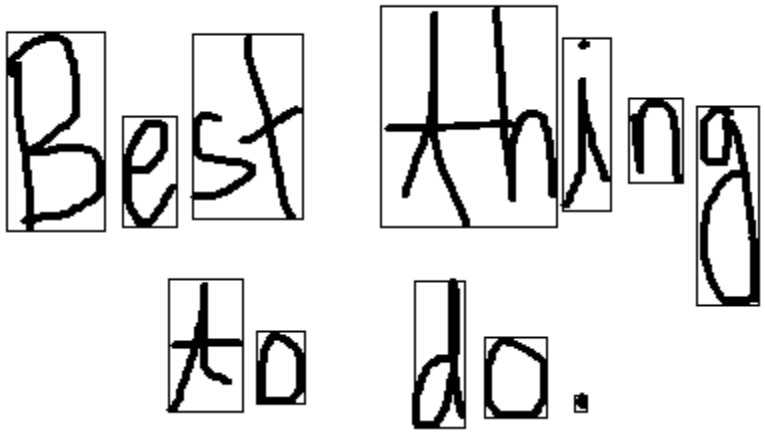


Figure 2.6 Connected components

There are two types of follow up processing. First one is boundary box analysis which is based on location and dimensions of each connected component and the second one is splitting of connected components.

2.4.3.1 Boundary Box Analysis

The boundary boxes do the proper segmentation of an image consisting of non-cursive characters. By testing their adjacency relationship to perform merging, or their size and aspect ratios to trigger splitting mechanisms, much of the segmentation task can be accurately performed at a low cost in computation.

This approach has been applied in segmenting handwritten postcodes [22] using knowledge of the number of symbols in the code. The connected components can be split according to rules based on the width and height of the boundary boxes (See Figure 2.7). Connected components are also used as segmentation of handwritten words. It is considered that words do not overlap but they can be fragmented.

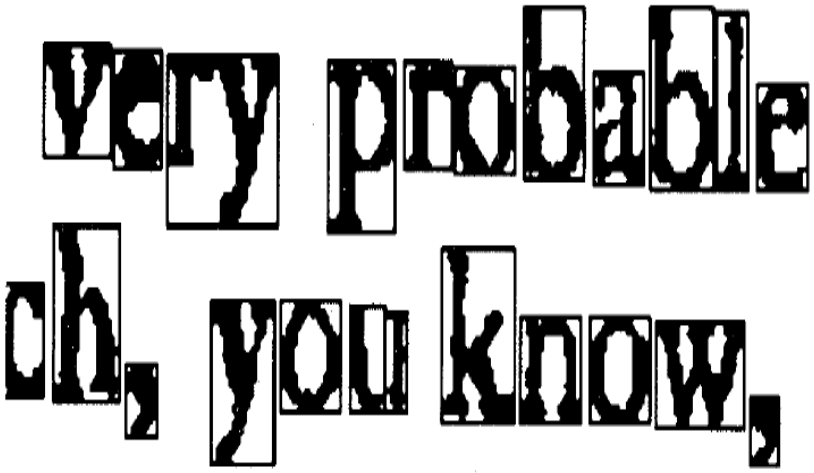


Figure 2.7 Boundary box Analysis

2.4.3.2 Splitting of Connected Components

Boundary Box analysis provides an efficient method to segments the non-touching handwritten characters. However there is some detailed processing is necessary in order to split the overlapped characters. The overlapping of two characters provides some special features. The dissection methods are used to detect these features and use these features to split the connected characters into segments (see figure 2.8).

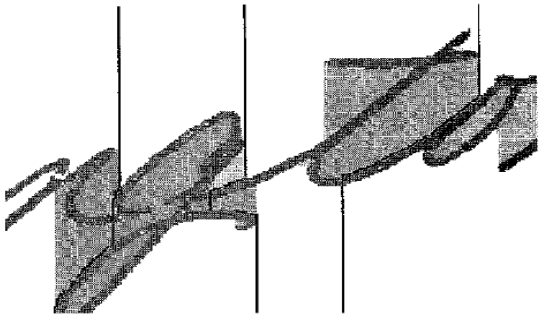


Figure 2.8 Splitting of Connected Components [22]

2.5 Binary Segmentation

In binary segmentation algorithm before segmentation we define the lower and upper boundary for the segmentation (see figure 2.9) and then decide the segmentation point at which we can segment the character as shown in the algorithm 2.1.

```
SET Upper Baseline = Calculate Upper Baseline()
SET Lower Baseline = Calculate Lower Baseline()
SET Vertical Pixel Density = Calculate Pixel Density (Upper Baseline,
Lower Baseline)
SET Stroke Width = Calculate Stroke Width()
SET Segment Criteria = Stroke Width * 2
SET Segment Points = Create Segmentation Points Array()
SET Index = 0
FOR EACH Vertical Pixel Density
    IF Vertical Pixel Density [Index] < Segment Criteria
        Segment Points[Index] = VALID
    ELSE
        Segment Points[Index] = INVALID
END FOR
WHILE (Continuous = Check Continuous Segmentation Points())
    Dissect Continuous into size of (Stroke Width)
END WHILE
```

Algorithm 2.1 Body region and Segmentation point selection [23]

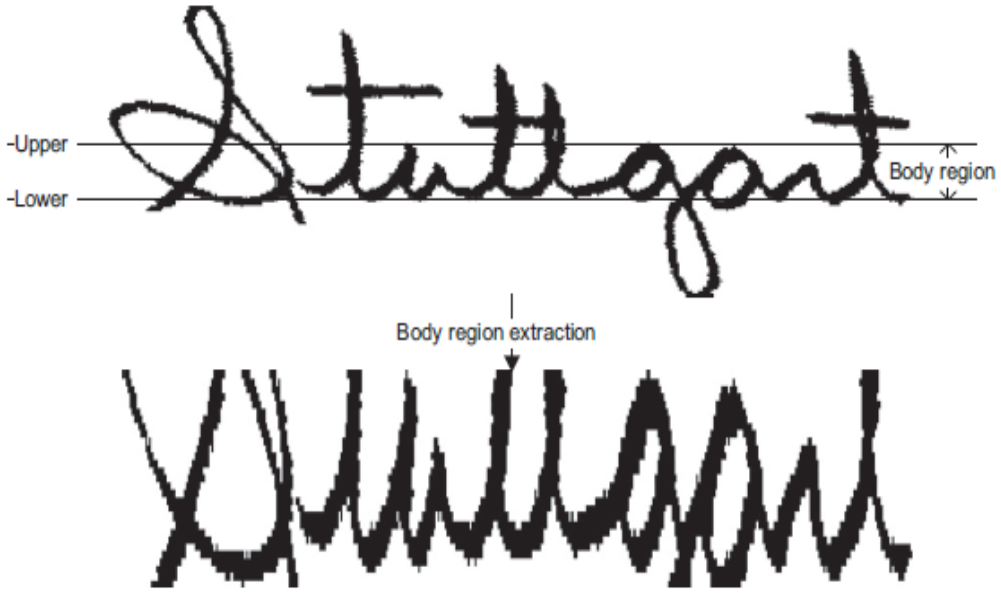


Figure 2.9 Lower and upper bound selection and Body region extraction [23]

After selecting the body region character is segmented as shown in the figure 2.10. The fig represents the segmentation tree. And then the next figure 2.11 represents the results of the segmentation tree.

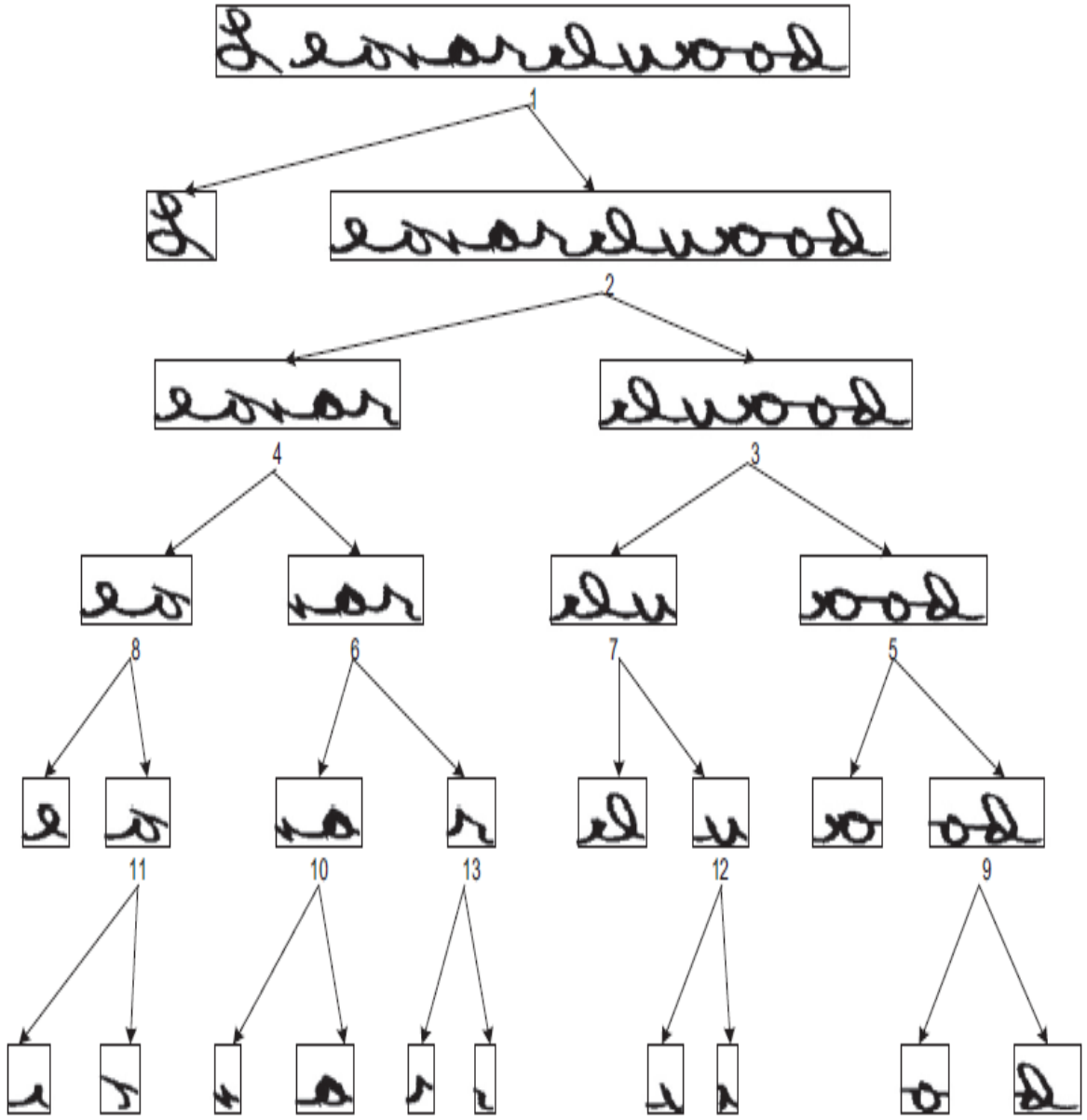


Figure 2.10 Segmentation Tree [23]

Figure 2.11 Resulting segmentation of the word after the segmentation tree

The overview of binary segmentation algorithm is presented in the figure 2.12.

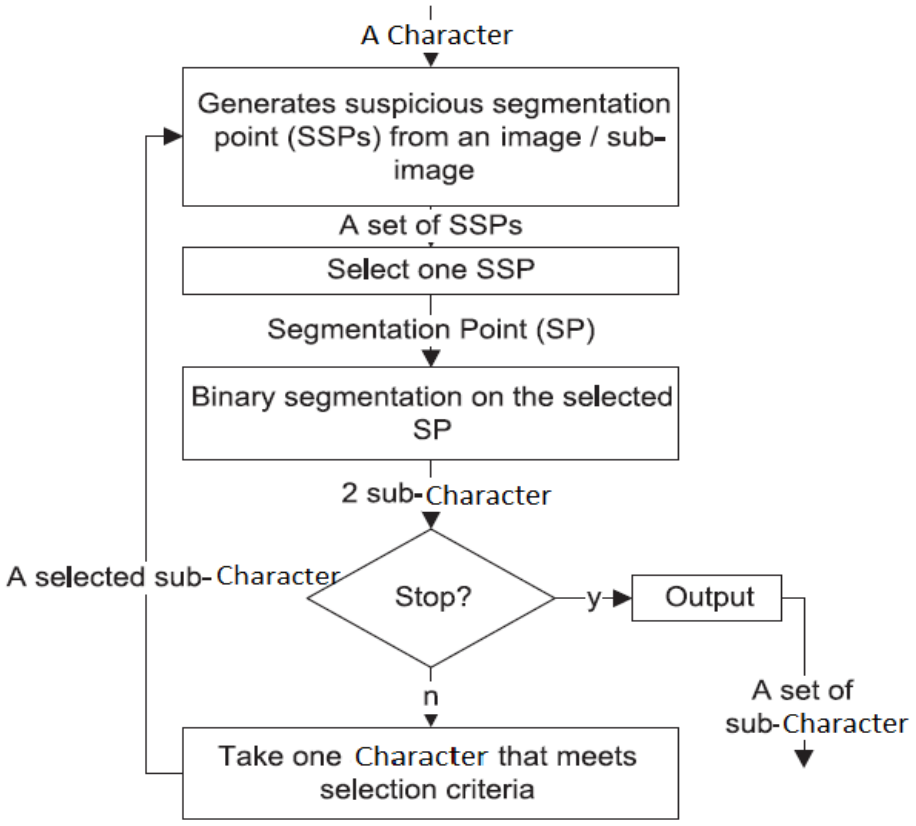


Figure 2.12 Representation of binary segmentation

As shown in the figure 2.12 in the binary segmentation approach a character is passed to the system and it generate suspicious segmentation points (SSPs). System generates the multiple sets of the suspicious segmentation points. Out of these suspicious segmentation points one set is selected and out of these sets one suspicious segmentation point is selected. This selected point used for the segmentation. On this selected point, system does the binary segmentation. Using this binary segmentation, a character is divided into multiple sub-characters and these sub-characters make a set.

Out of these sets, one set of sub-character is selected and out of that set one sub-character is selected, which can be further segmented, and this sub-character is again passed to the system for further segmentation. This repetition is done since we get the sub-characters which can't be segmented. After the segmentation we get a set of segmented sub-characters.

2.6 Hidden Markov Models (HMM)

Hidden Markov Model (HMM) is a widespread method for handwritten character segmentation. The Hidden Markov model represents state to state transitions within a character. These transitions provide a sequence of observations on the character. Features are typically measured in the left-to-right direction. In this model segmentation is done by matching the given sequences of the feature to the model. In Hidden Markov Model, to choose the most appropriate word from a set, the designer of model may select either the composition model or else model which maximize the probability of the observation.

Every letter does not have the different boundaries and for mapping that fact the HMM is very powerful tool. Generally the perfect letter dissection can't be achieved and this problem can be handled by using HMM because they learn by the observing letter segmentation on a training set.

2.7 Line terminal and Shape feature

In feature extraction phase system finds the feature of the given input. System finds the shape, size and boundary of the given input. System calculates the line terminals and shape features (see figure 2.13). In the shape features system calculates the cusps

points, bumps points, loop points, angle points and cross points. With the help of these various points system finds the shape of the given handwritten character. These shapes help of the system to segment the handwritten character. These features are collected and used for the training of the system. These features also help for the stroke identification.

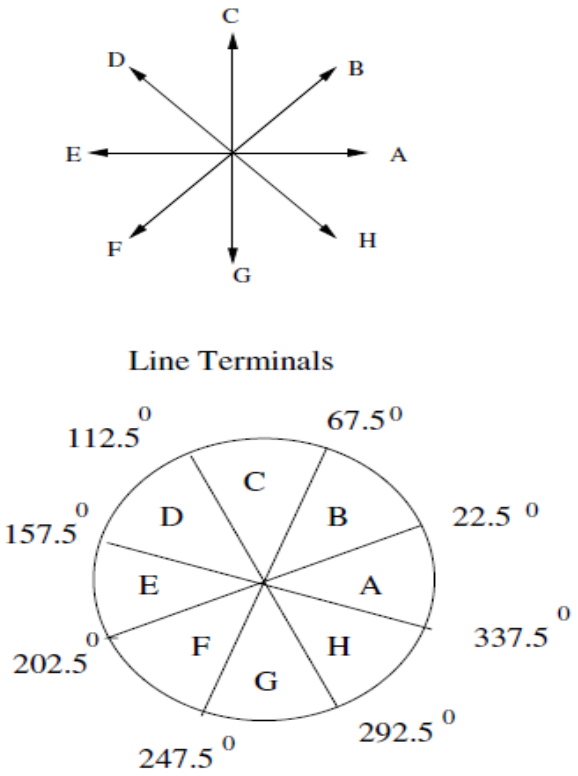


Figure 2.13 Line Terminals and Directional Coding

Cusps: A point where derivation of x with respect to time and derivation of y with respect to time simultaneously go to zero is called cusp (R).

Loop point: A point where a loop starts and end called the loop point, and loop is a point where a point of the stroke and end point is same.

Cross-point: Where two strokes intersects each other a cross point exist.

Angle point: An angle point is exists between two strokes.

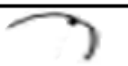
Cusp points, loop points and cross-points are known as critical points. Table 2.1 represents the critical points.

Table 2.1: Critical points

Feature	Description	Example
R	Cusp point	
S	Loop point	
X	Cross point	

Bumps point: A points where any tangent exist and on side of this tangent is wholly is called the bumps point. Table 2.2 represents the bumps points.

Table 2.2: Bump points

Feature	Description	Example
J	X-bump, +ve Curvature	
K	X-bump, -ve Curvature	
L	Y-bump, +ve Curvature	
M	Y-bump, -ve Curvature	
N	+45°-bump- +ve Curvature	
O	+45°-bump- -ve Curvature	
P	−45°-bump- +ve Curvature	
Q	−45°-bump- -ve Curvature	

When the preprocessing is done on the given stroke data then either system find the 8-Directional code feature or find the shape feature as shown in the figure 2.14. In case of 8-directional code, system detects the line terminal. After detecting the line terminal, system creates a string of these detected line terminals.

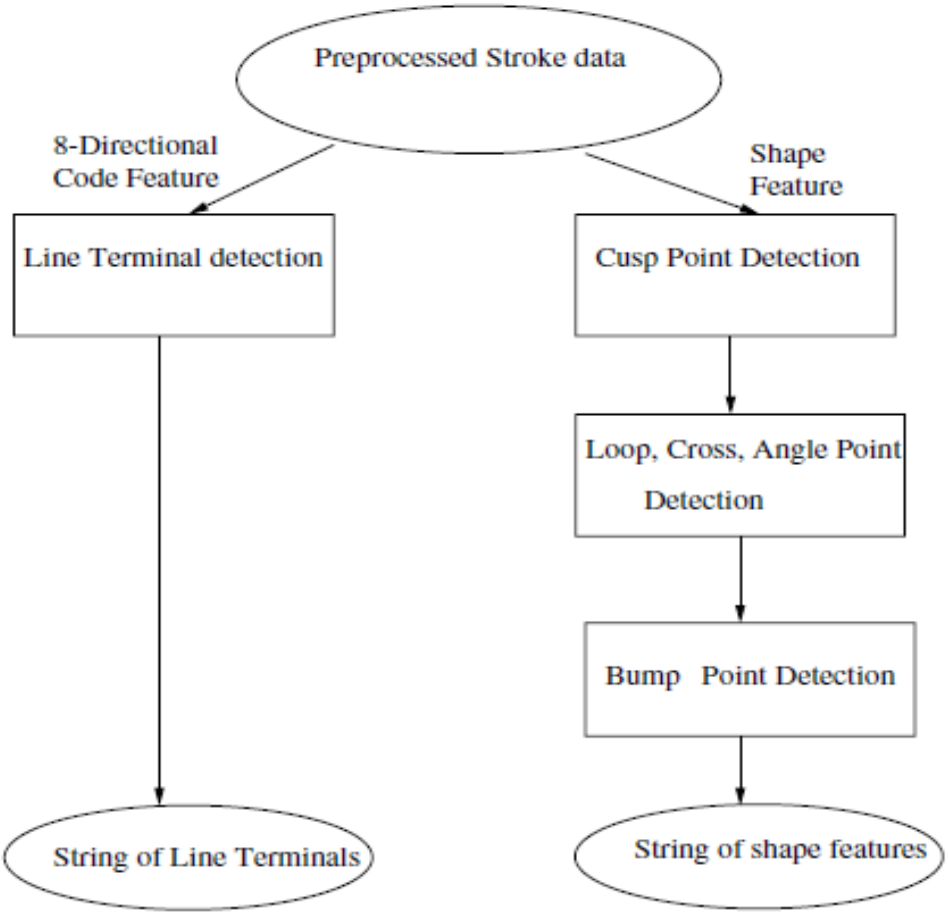


Figure 2.14 Feature Extraction Process Flow Diagram

When system works on the shape feature, system detects the cusps points and then loop, cross and angle points. And then it detects the bumps points. After detecting these shape features, system creates a string of these detected shape features. This queue is used for the detection of the handwritten character.

Chapter Summery

In this chapter, various techniques which are used for the handwritten character segmentation are represented. Most of the methodologies which are used for the handwritten character segmentation are based on the shape identification. Since there are highly similarities between the characters then these methodologies does not work perfectly for every language. In this chapter some famous work done on handwritten character segmentation like Nebulous Stroke Model, Elastic Matching, Analysis-by-synthesis, Dissection Techniques for Segmentation, Binary Segmentation, HMM, Line terminal and Shape feature are represented very briefly.

Chapter 3

Problem Statement

There are several tools available for taking a handwritten input from the user and segment the handwritten characters. There are also several problems associated with these tools which are follows:

- Various tools which are available for the taking input from the user and segment these are machine dependent and there is not any tool available which works on the all type of systems and platforms.
- Every tool need to install some type of framework or application the system for the working.
- Most of the tools are language dependent so they need to install these language frameworks on the system.
- There are some technologies that work upon the concept of n-gram model which work on the concept of probability, i.e. the probability of occurrence of a character when another character has already occurred, most of the time does not work correctly.
- Tools which are available for the segmentation, segments the handwritten character according to their shapes and then provide these shapes to the recognition machine to recognize these shapes but most of the times these segments create the ambiguity for the system to recognize these shapes, because most of the characters shares the same segments.
- There are several difficulties involved in segmenting a character into the sub-character. Another problem is that it is not clear how to break a letter into sub-character units.
- Sometimes a character can be over segmented and then these segments can make a different character. When a character is over segmented then one of its segment may become another character, this is called the over segmentation. Once a character is over segmented, it cannot be recovered and cannot be recognized.

Developing the Online Handwritten character Segmentation system has some greater challenge because of the following reasons:

- **Presence of very large number of character sets**

  Since there are very large numbers of data sets because of very large number of characters in the different-different languages then it is very difficult to properly segment them. There are so many languages and these languages have so many characters. These characters make a huge data set which creates the problem for the system for segmenting these characters.

- **High degree of similarity of shapes between the character**

  In the characters set, there are so many characters which have the fame features like shapes and any other. Due to similar shapes it is very difficult to segment the character perfectly and recognize the desire character. When system tries to segment the character which has similar shapes then it creates the ambiguity for the system. This ambiguity creates the problem in the proper segmentation of the handwritten character.

Chapter 4

Objective and Methodology

Generally humans can write faster than they can type on the computer. So there is a need to add a methodology by which humans can give a command to the system through handwriting. For that requirement, there is a need to create handwritten character recognition system. Across the globe there are billions of users which can use this system. Since different users has different languages and different handwriting styles and speed, there is a need to create an input tool that can take input in any handwriting style and can properly segment them.

To make a platform independent tool, it is important that the tool must support and able to accept all handwritten characters and accept it. The most challenging part is to recognize the characters of different and uneven user's handwriting.

4.1 Objectives

The main objective of research work is to create a tool which is able to accept all handwritten characters of different users and match it with training set of alphabets of different languages. In order to perform this task, following objectives are proposed to be carried out.

i) To create a platform independent tool for taking handwritten input

ii) The tool should be able to capture and process the handwritten input for all languages.

iii) To segment and capture different strokes.

4.2 Methodology

To achieve the objectives discussed in section 4.1, HTML5 is used.

i) To achieve the first objective, canvas is developed using HTML5. Canvas provide feature to store the relevant set of coordinates (x,y). Each pair of coordinates is stored in an array so that the manipulation of coordinates became easy.

ii) Different strokes are captured in different array. Since we are getting the co-ordinates of the canvas where user pens down or mouse down to pens up to mouse up, we can extract the feature as there is any change in direction of the writing.

iii) We have created a window as the canvas window created earlier to show the normalized handwriting. Since we have not restricted the user to write in particular area of the canvas. So it was important that the user handwriting is pre-processed as normalized handwriting.

For this purpose, the height of the normalized window is fixed *i.e.* 150 pixels in our case. This was important because uneven height may produce problem while matching with the characters from the database. Since only left to right writing languages have been considered in our case, thus we didn't restricted area for writing running in x-direction. It benefits us to divide the normalized window in more number of segments to match it against the characters stored in database.

iv) To develop data set of different characters of different languages the handwritten characters from the normalized window have been encoded and converted into 64bit encode. Then that encoded text is to be transfer in image. Manually the user has to save the image to be saved for the character matching through matcher. That image is collected in a folder from where one by one it is called and reverse process is performed to finally match and get that character segmented from the handwriting.

v) According to the width of the called character from the dataset, the normalized window is divided along width. Pixel by pixel is matched between both the windows. If the pixel matched is above the certain amount of the pixels then the character is found. According to the assumption, the segment having maximum pixel matching represents the found of called character in the segment of the normalized window. If the segment is not matched or found, the width is increased by 10 pixels to the right and the process is repeated again.

Chapter 5

Implementation

There are several applications and techniques available for the segmentation of a handwritten character but there is a major problem that how to break a handwritten character into multiple parts so that these parts can be passed to the recognition system for the recognition of that handwritten character.

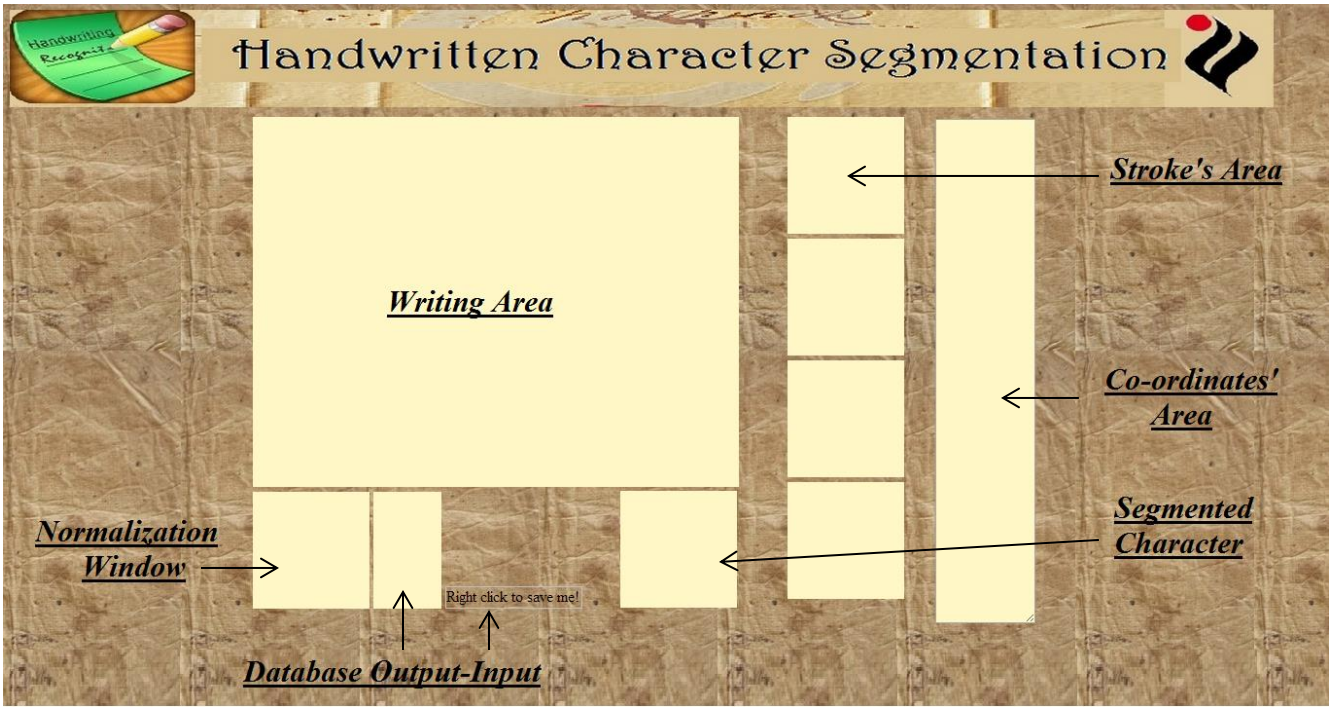


Figure 5.1 Components of the handwritten character segmentation tool

For solving these types of problems, which are mentioned above, a new segmentation methodology is proposed. In this methodology the handwritten characters are not segmented on the basis of their shapes instead of that the handwritten characters are being taken on a platform by a touch devices like any PDA device, any touch pad, mobile or tablets and then the co-ordinates of the each point, where pen or stylus moves from pen down to pen up, is being taken. The interface of this tool is represented on the figure 5.1.

As shown in the above diagram, there is a writing area where user writes the character. There is stroke's area, each stroke area shows a stroke by which that character is written by user. There is also a co-ordinate area where co-ordinates of the written character are shown according to their strokes. There is a normalization window where handwritten characters get normalized. There is database output-input window, input window is used to store a character in the database for the training and output window shows the character which is matched to the handwritten input and this character is taken from the characters' database. Figure 5.2 represents the working of the handwritten character segmentation system.

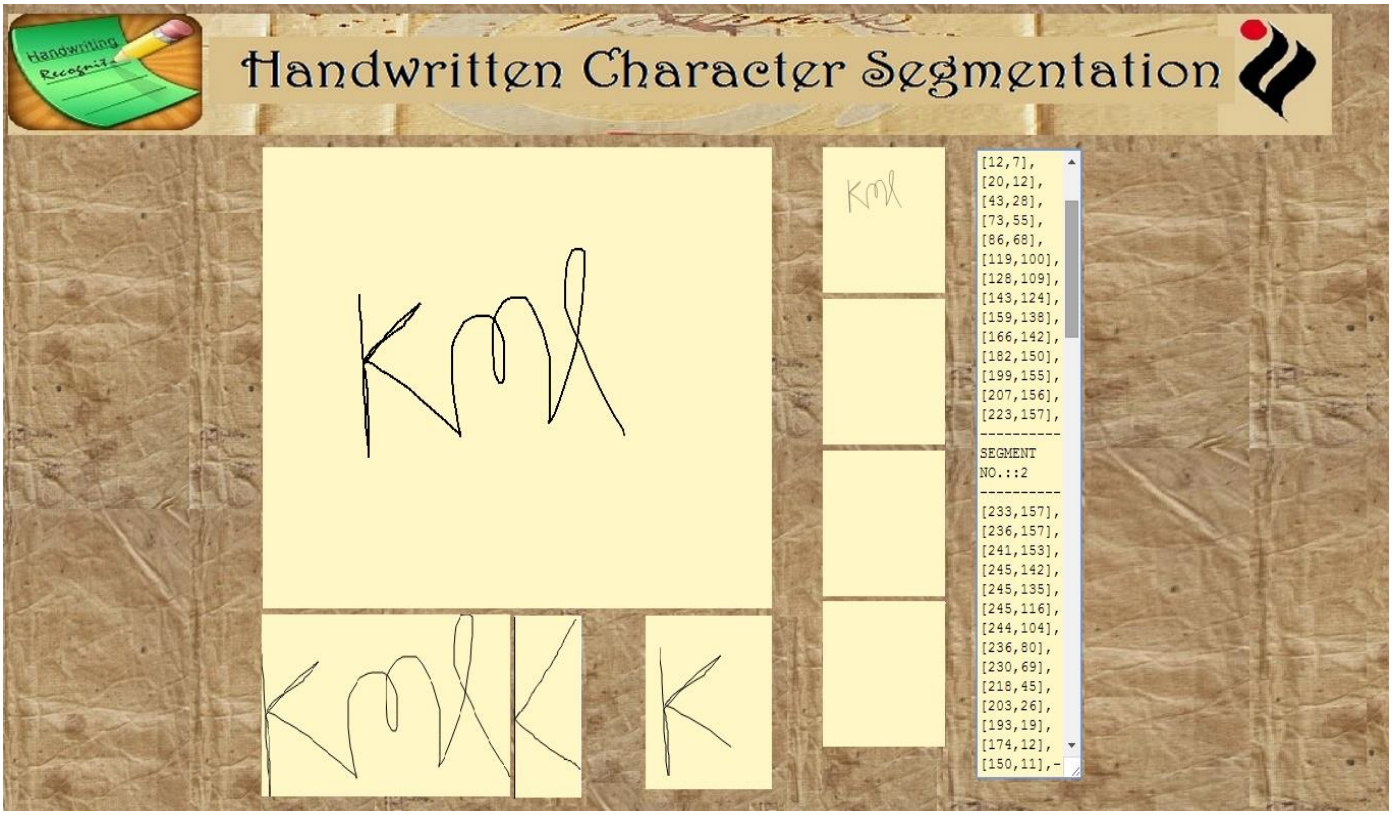


Figure 5.2 Working of the handwritten character segmentation tool

5.1 Architecture of Segmentation tool

Architecture of handwritten character segmentation is shown in the figure 5.3. The architecture of handwritten character segmentation tool contains the Handwritten

input, Normalization, Character Database, Feature extraction, Indexing, Character matching and Output phase, which are described below.

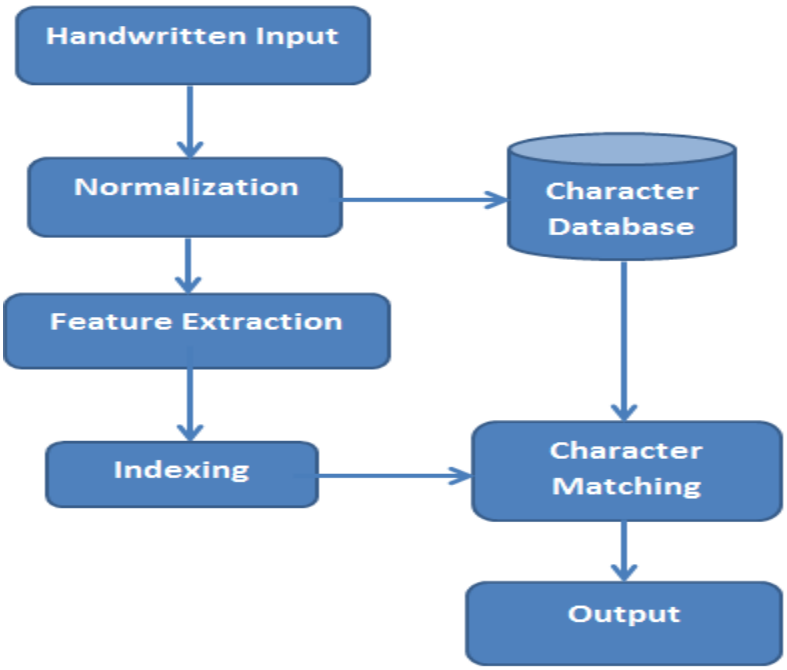


Figure 5.3 Architecture of handwritten character segmentation

5.1.1 Handwritten Input

In this system a writing area is created with the help of canvas tool of HTML5. This writing area is used for taking handwritten input from the user. User gives the handwritten input on this writing area using any touch sensitive devise like Mobile, Tablet, and PDA etc. It uses the two functions pen up and pen down. When user touches the device using any pen, stylus or finger then it is called the pen down and the moment when he disconnect the pen to the device then it is called the pen up. The movement of the pen from pen down to pen up is called a stroke. User can either write a single character into multiple strokes or multiple characters into single stroke.

5.1.2 Normalization

User writes the input on the writing area and then this input is normalized into the normalized area. Figure 5.4 represents the flow diagram of the normalization of the handwritten character which is given as input by the user on the writing area.

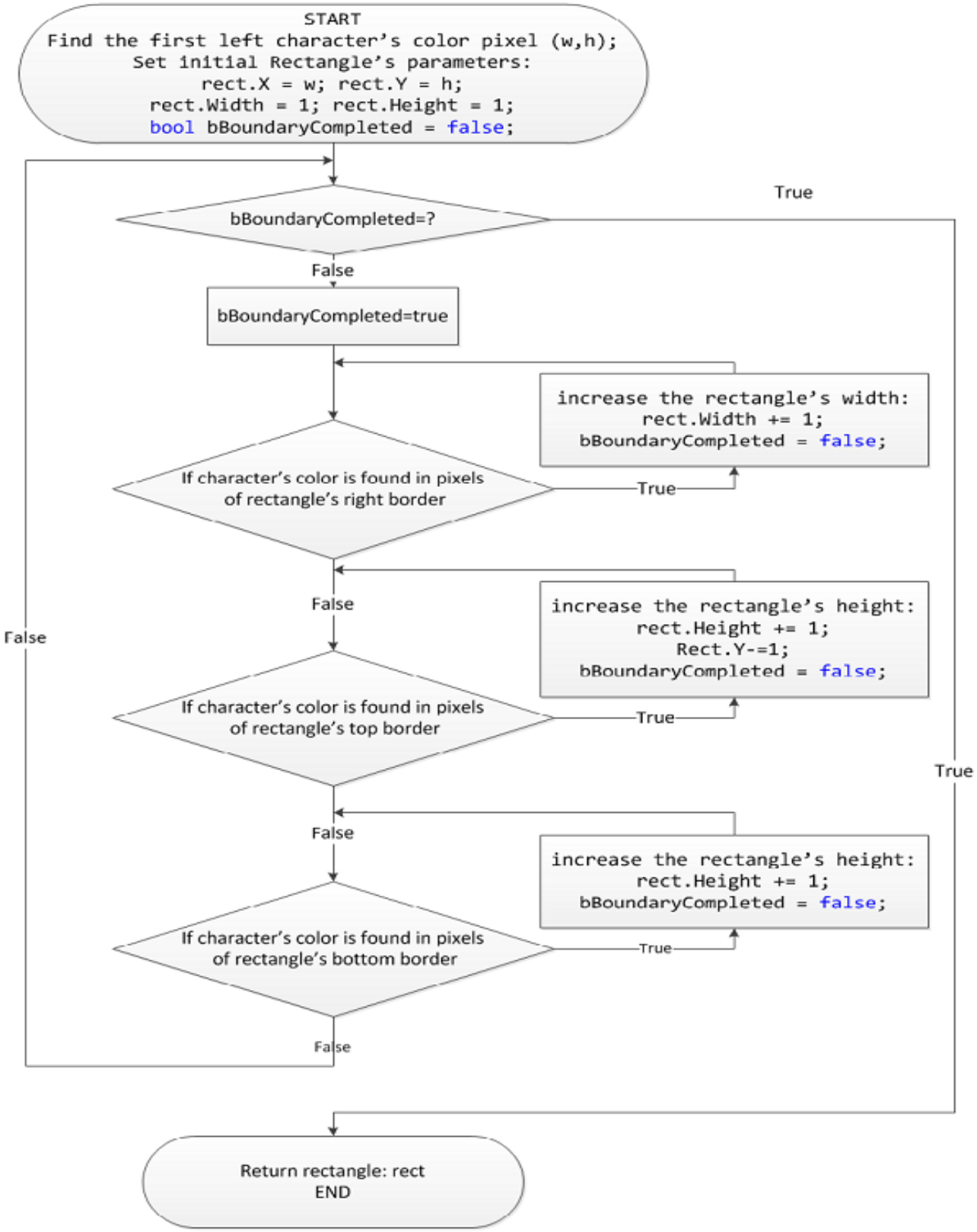


Figure 5.4 Flow diagram of normalization

As shown in the figure 5.4, at first system finds the left most character and then the first color pixel and from this color pixel, system start to create a normalize window. The creation of boundary box is also shown in the figure 5.5. The boundary is created from left to right until system finds the last rightmost pixel of the input and then similarly it finds the boundary from top to bottom. When system finds the left most and rightmost pixel it creates a boundary from left to right and when it finds the topmost and bottommost pixels, system also creates the top to bottom boundary and in this way it creates a rectangle boundary of the handwritten input.

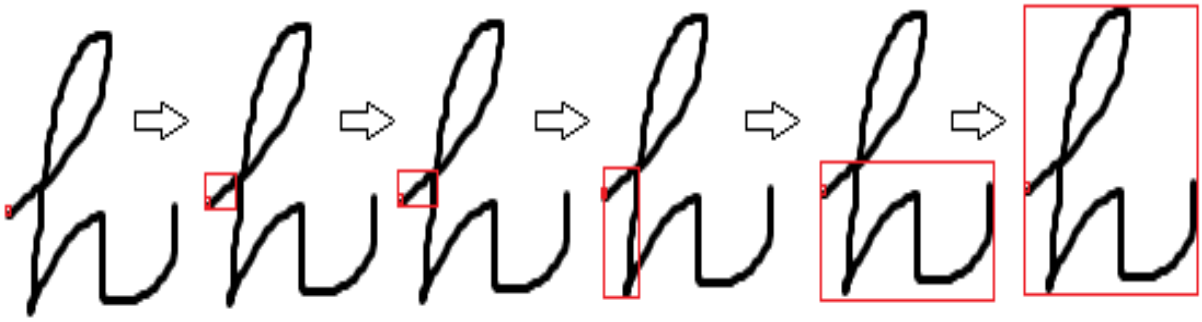


Figure 5.5 Boundary box creations

Let's consider that the first pixel in the x axis is Xs and the last pixel in the x axis is XL similarly the first pixel on the y axis is Ys and the last pixel on the y axis is YL. For calculating the total area a formula is used.

$$\text{Length of X axis} = X_L - X_S$$

Similarly,

$$\text{Length of Y axis} = Y_L - Y_S$$

This is the area of variable window size. Now this variable window will be converted into fixed size window. This fix sizes window is called Normalization window. Figure 5.6 represents the normalization of the handwritten user input from the formulas which are described above.

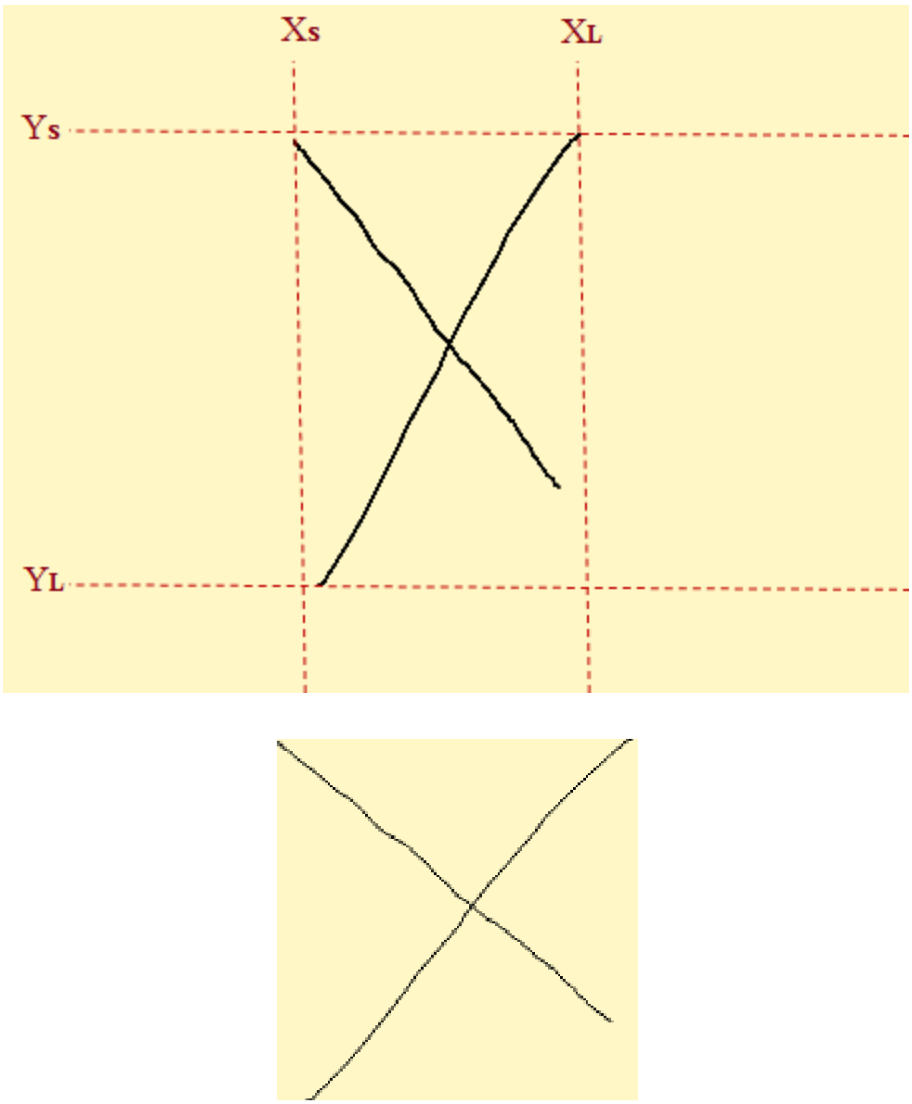


Figure 5.6 Normalization of Handwritten character

5.1.3 Character Database

After normalized the character, this character can be saved into the character database so that we can further segment any character on the basis of that character. There is a database input window by which we can save any character into the database. As shown in the figure 5.7 we can save the normalized character into the database. Character database contains all the characters which are stored from the database input window. Lately these characters are used for the segmentation of the similar characters.

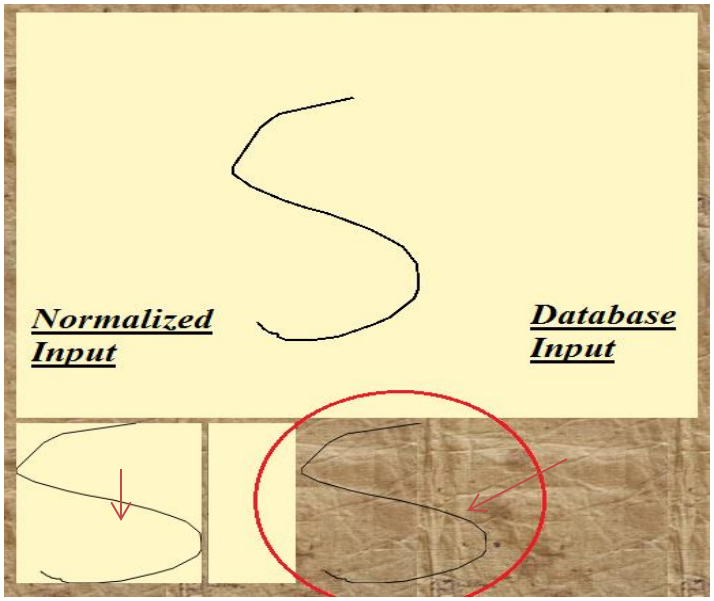


Figure 5.7 Database Input

5.1.4 Feature Extraction

After the normalization of the handwritten input, all the pixel co-ordinates of the input are calculated and that are used for the single character segmentation. If a character is written into multiple strokes then these strokes are shown in the stroke's area as shown in the figure 5.8.

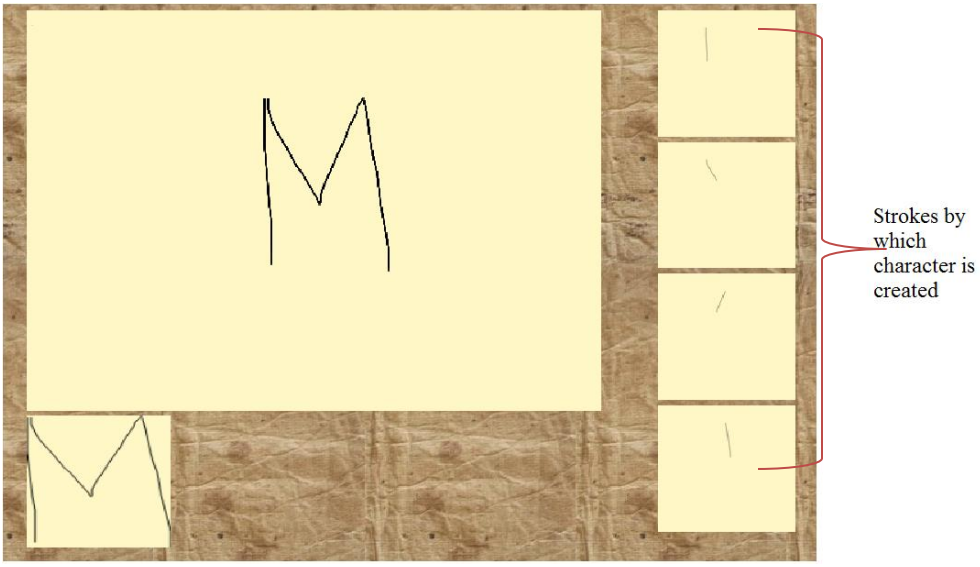


Figure 5.8 Single character's strokes

If user write the single character into single stoke then it will also give the number of strokes by which this character can be created. These numbers of strokes we can see in the co-ordinates area. A character will be generated by multiple strokes and out of them some will be discarded. All the discarded and actual number of strokes will be present in the strokes area as shown in the figure 5.9.

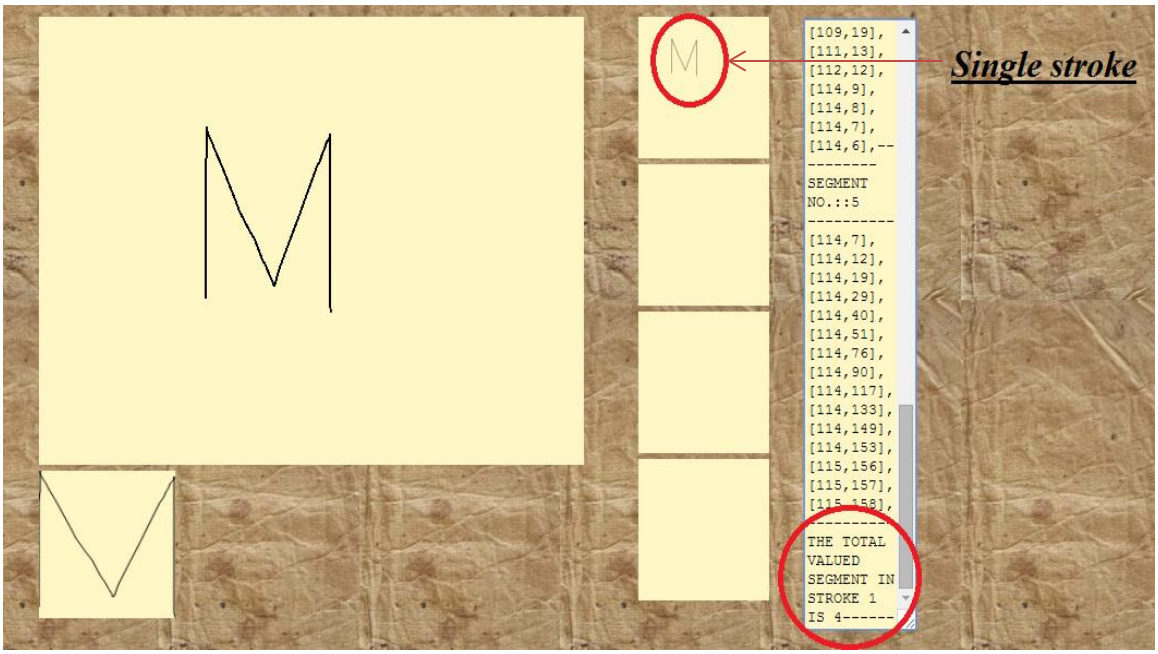


Figure 5.9 Single character segmentation

The co-ordinates which are represented into the co-ordinates' area are calculated from the normalized window. Let's consider the co-ordinates of x axis of the user input on the writing area are stored in the array a[i] and co-ordinates of the y axis of the user input on the writing area are stored in the array b[i].

Size of variable window is Wb and size of fix normalized window is Ws. Height of the variable window is Hb and height of fix normalized window is Hs.

Then,

The co-ordinates of x axis on the fix normalized window

$$X[i] = [a[i]-Xs]\times\frac{Ws}{Wb}$$

The co-ordinates of y axis on the fix normalized window

$$Y[i] = [b[i]-Ys]\times\frac{Hs}{Hb}$$

There is also an xml file can be created from these co-ordinates and it can be compared to the standard co-ordinates of that character and in this way it is also may use for the segmentation and recognition of the handwritten character segmentation. The standard co-ordinates of a character are shown in the fig 5.10.

```
<?xml version="1.0" encoding="UTF-8"?>
- <wordSetDef>
  - <word>
      <wordNo>80</wordNo>
      <totalStrokes>2</totalStrokes>
    - <wordDesc>
        X
      - <stroke>
          <strokeNo>1</strokeNo>
          <strokeId>379</strokeId>
        - <point>
            <X>165</X>
            <Y>160</Y>
          </point>
        - <point>
            <X>176</X>
            <Y>168</Y>
          </point>
        - <point>
            <X>185</X>
            <Y>174</Y>
          </point>
        - <point>
            <X>194</X>
            <Y>180</Y>
          </point>
        - <point>
            <X>203</X>
            <Y>188</Y>
          </point>
        - <point>
            <X>210</X>
            <Y>192</Y>
          </point>
        - <point>
            <X>216</X>
            <Y>196</Y>
```

Figure 5.10 Standard co-ordinates of the character

5.1.5 Indexing

After the feature extraction we get so many strokes of a character out of which some are not desired. In indexing, the system discards the undesired strokes and only keeps the co-ordinates of the strokes which are desired. It calculates the number of strokes and then display to the end of the stroke. Because of indexing we can calculate that a character will be segmented into how many segments and that helps in the handwritten character segmentation.

5.1.6 Matching

A handwritten input character is matched from the character database in the matching phase. After Normalization, let the window be Nw. The character to be matched comes one by one in called temporarily in window Mw. The Algorithm 5.1 represents the working for matching.

Algorithm 5.1

1. Start
2. Declare TotalPixelMatched=0; compare=0; i=0; m=0;
3. Mw= call alphabet to be matched
   //get image data of window to be matched, Mw
4. Imagedata1←getImageData (0,0, Mw.width, 150)

   Data← imagedata1.data

5. Repeat until i <= Nw. width
   TotalPixelMatched=0
   //get image data of handwriting character
   5.1 image data 2← get image data (i,o, Mw. Width,150)
   data2 ←imagedata2. Data

   // count total pixel matched in segment.

   5.2 repeat until m < data2. length
   { if data1 [m]==data 2[m]

TotalPixelMatched ++}

5.3 if (TotalPixelMatched >= 52,000 && TotalPixelMatched > compare

{compare== TotalPixelMatched

// Get imagedata of matched segment

// Replaced if matched earlier

imagedata←get Image Data (i,0,Mw.width,150)

Data←imagedata.data}

5.4 i+=20

6. Put image data of matched segment on output window

   Put Images (imagedata,0,0)

7. Stop

5.1.7 Output

If the shape of the handwritten input character is matched with the shape which is stored in the database then that character is recognize from the input and then segmented as shown in the figure 5.11.

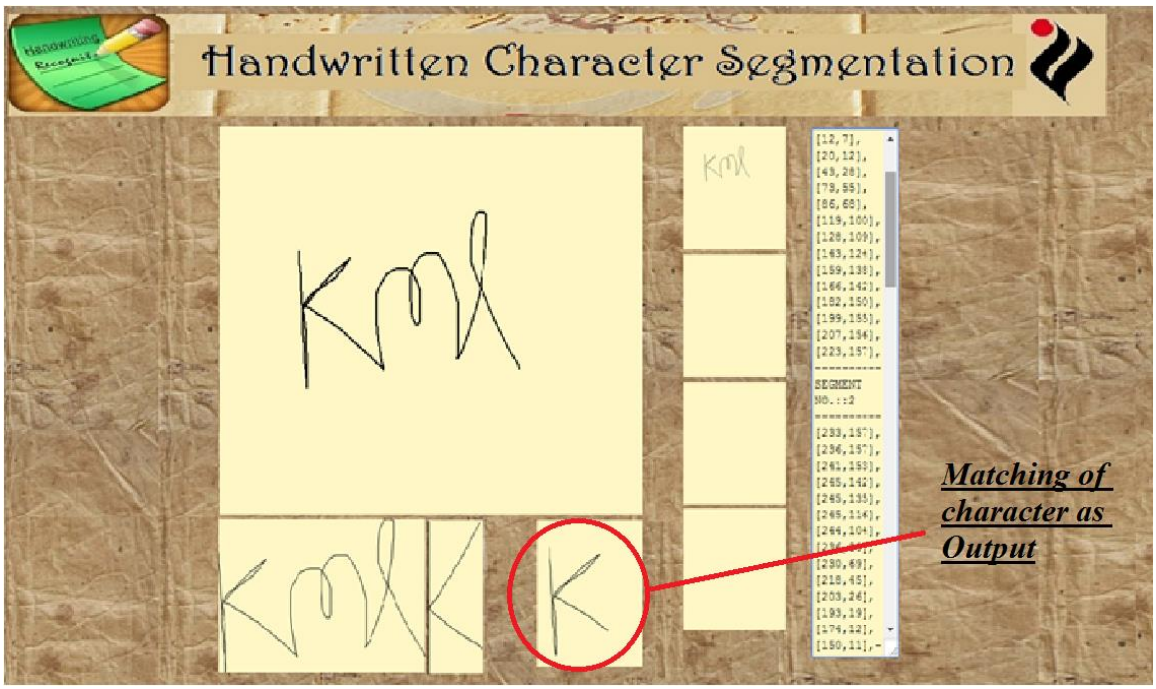


Figure 5.11 Output of the segmentation tool

As shown in the above figure, a user gives “kml” as the handwritten input and after all the process the first character is matched with the character “k” which is store in the database so the character “k” from the input is recognized and segmented which is shown in the output.

Chapter Summery

This chapter represents the work done for creating a universal handwritten character segmentation tool. Most of the tools which are available for the handwritten character segmentation are either machine dependent or shape recognition based. In this chapter the implementation of the proposed work is represented. In the represented methodology, user gives the handwritten input on the writing area and then that input is normalized into the normalization window. System finds the co-ordinate of that normalized input from the normalization window. These co-ordinates are used for the segmentation of the character and also used for the recognition of the character. This tool is runs on the all browsers which are HTML5 enable. Since now days most of the browsers are HTML5 enable then it is a universal tool for taking an input from the user and then segment it.

Chapter 6
Conclusion

Tools which are available for the segmentation segment the handwritten character according to their shapes and then provide these shapes to the recognition machine to recognize these shapes. The tools which recognize the characters on the basis of their shapes cannot work correctly. They create the ambiguity for the system. For solving these types of problems a new segmentation technique is proposed in this report. In this technique the handwritten characters are not segmented on the basis of their shapes instead of that user draw the character on an interface and system segments that character. System finds the co-ordinates of the character which is drawn by the user. These co-ordinates can be used in character recognition or in shape recognition. If we use the co-ordinates for the recognition of any character then they can be recognized without ambiguity.

During recent years, the online handwriting recognition has become a very important task in every day applications. Characters are written left to right and top to bottom. Generally a writer writes the words in which characters are connected to each other. When characters are connected to each other they create the different-different shapes which are not easy to segment. So there was need of a different technology to segment any handwritten character which can segment the character vary efficiently and this technology is proposed in this project.

The handwritten character input and segmentation tool is an online tool. This tool is based on the HTML5. HTML5 is a hypertext markup language which is used for structuring and presenting content for the W3 (World Wide Web) and a core technology of the Internet. HTML5 is the 5th revision of the HTML standards. This tool is constructed using HTML5 and java script.

Since this tool is constructed using HTML5, it is a universal tool. This tool can run on any browser which has HTML5 enable. Now days every browser is HTML5 enable so it is a universal tool for the segmentation of the handwritten character. Since it

made by the HTML5 then it is also supported by the standard Mobile and Tablet devices. In this project the Canvas tool of HTML5 is used. When user gives an input then it is written on the canvas tool and it converts that input into co-ordinates.

User gives the input on the writing area and system normalizes that handwritten input. After normalizing the input system finds the co-ordinates of the pen movement. From pen down to pen up on the canvas or touch sensitive screen, all the movements are captured from the system and these movements are converted into the co-ordinates on which pen moved. All of these co-ordinates are used for the segmentation of the single character. Remaining all canvases which are placed into the stroke area used to represent the strokes which are drawn by the user of the system. On these canvases we represent the strokes by which a character is drawn.

References

[1] Hidetoshi Miyao and Minoru Maruyama “Writer Adaptation for Online Handwriting Recognition System Using Virtual Examples” Shinshu University, 4-17-1 Wakasato, Nagano 380-8553, Japan 2009 10th International Conference on Document Analysis and Recognition.

[2] T. Artieres and P. Gallinari. Stroke level HMMs for online handwriting recognition. In IWFHR ’02: Proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition (IWFHR’02), page 227, Washington, DC, USA, 2002. IEEE Computer Society.

[3] T. Artieres, S. Marukatat, and P. Gallinari. Online handwritten shape recognition using segmental hidden markov models. IEEE Trans. Pattern Anal. Mach. Intell., 29(2):205–217, 2007.

[4] Rejean Plamondon and Sargur N. Srihari “On-Line and Off-Line Handwriting Recognition” A Comprehensive Survey 1EEE Transactions on pattern analysis and machine intelligence. vol. 22, no. 1.pp. 63-84 january 2000

[5] S.N. Srihari, "High Performance Reading Machines," Proc. IEEE vol. 80, no. 7, pp. 1,120-1,132, 1992

[6] P. Ahmed and C. Y . Suen, ”Computer recognition of totally unconstrained handwritten Zip codes,” Int. J . Pattern Recognition Artificial Intell. , vol. 1, pp.1-15, 1987.

[7] A. L. Knoll, “Experiments with characteristic loci for recognition of hand printed characters,” IEEE Trans. Comput., vol. C-18, pp. 366-372, Apr. 1969.

[8] P. W. Becker and K. A. Nielsen, “Pattern recognition using dynamic pictorial information,” IEEE Trans. Syst., Man, Cybern., vol. SMC-2, pp. 434-437, July 1972.

[9] S. A. Guberman and V. V. Rozentsveig, “Algorithm for the recognition of handwritten text,” Avtomarika i Telernekanika, vol. 5, pp. 122-129, May 1976.

[10] Schantz, Herbert F. (1982). The history of OCR, optical character recognition. [Manchester Center, Vt.]: Recognition Technologies Users Association. ISBN 9780943072012.

[11] R. Nag, K.H. Wong, and F. Fallside, "Script Recognition Using Hidden Markov Models," Proc. TCASSP '86, vol. 3, pp. 2,071-2,074, Japan, Apr. 1986.

[12] Jianying Hu, Michael K. Brown and William Turin, "HMM Based On-Line Handwriting Recognition" IEEE Transactions on pattern analysis and machine intelligence, vol. 18, no. 10, pp. 1039-1045 October 1996.

[13] J. B. Kruskal, "An overview of sequence comparison: Time warps, string edits, and macromolecules," SIAM Rev., vol. 25, pp. 201-237, Apr. 1983.

[14] D. Sankoff and J. B. Kruskal, Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison. London: Addison-Wesley, 1983.

[15] C.C. Tappert, "Cursive Script Recognition by Elastic Matching," IBM J. Research Development, vol. 26, pp. 765-771, Nov. 1982.

[16] J. J. Denier van der Gon, 1. P. Thuring, and J . Strackee, "A handwriting simulator" Phys. Med. Bioi., vol. 6, pp. 407-414. 1962.

[17] J. J. Denier van der Gon and J . P. Thuring, "The guiding of human writing movements," Kybernetik, vol. 2, pp. 145-148, Feb. 1965.

[18] "Handwriting generation and recognition," in P. A. Kolers and M. Eden, Ed., Recognizing Patterns. Cambridge, MA: M.I.T. Press, 1968, pp. 138-154.

[19] "Handwriting and pattern recognition," IRE Trans. Inform. Theory, vol. IT-8, 1962.

[20] R.L. Hoffman and J.W. McCullough, "Segmentation Methods for Recognition of Machine-Printed Characters," IBM I. Research and Development, pp. 153-65, Mar. 1971.

[21] H.S. Baird, S. Kahan, and T. Pavlidis, "Components of an Omni- font Page Reader," Proc. Eighth Int'l Conf. Pattern Recognition, Paris, pp. 344-348,1986.

[22] M. Cesar and R. Shinghal, ”Algorithm for Segmenting Hand- written Postal Codes,” Int’l J. Man Machine Studies, vol. 33, no. 1, pp. 63-80, July 1990.

[23] L. Hong, and B. Verma. "Binary segmentation algorithm for English cursive handwriting recognition." Pattern Recognition 45.4 (2012): 1306-1317.

List of Publication

- Mayur Vyas, Karun Verma, "A Platform Independent Methodology for On-line Handwritten Character Segmentation" in International Conference on Advanced Communication Control and Computing Technologies (ICACCCT) (IEEE), ISBN No. 978-1-4799-3914-5/14/2014, IEEE, pp 1480-1483.
- Mayur Vyas, Karun Verma, "A Comprehensive Survey of Handwritten Character Segmentation" in International Conference on Advanced Communication Control and Computing Technologies (ICACCCT) (IEEE), ISBN No. 978-1-4799-3914-5/14/2014, IEEE, pp1462-1465 .