ISLAMIC UNIVERSITY OF TECHNOLOGY

# OFFLINE BANGLA HANDWRITTEN CHARACTER RECOGNITION

*Authors*
**Yamin Bin Islam (104405)**
**Md. Taiyebur Rahman (104412)**

*Supervisor*
**Md. Hasanul Kabir, Ph.D.**
**Assistant Professor**
**Department of Computer Science and Engineering**

**A thesis submitted to the Department of CSE**
**in partial fulfillment of the requirements for**
**the degree of B.Sc. Engineering in CSE**
**Academic Year: 2013-2014**

A Subsidiary Organ of the Organization of Islamic Co-operation
Gazipur, Bangladesh

October 2014

# Declaration of Authorship

This is to certify that the work presented in this thesis is the outcome of the analysis and investigation carried out by Yamin Bin Islam and Md. Taiyebur Rahman under the supervision of Dr. Md. Hasanul Kabir in the Department of Computer Science and Engineering (CSE), IUT, Gazipur, Bangladesh. It is also declared that neither of the thesis nor any part of this thesis has been submitted anywhere else for any degree or diploma. Information derived from the published and unpublished work of others has been acknowledged in the text and a list of references is given.

*Authors:*

______________________________________

Yamin Bin Islam

Student ID: 104405

______________________________________

Md. Taiyebur Rahman

Student ID: 104412

*Supervisor:*

______________________________________

Md. Hasanul Kabir, Ph.D.

Assistant Professor

Department of Computer Science and Engineering

Islamic University of Technology (IUT)

# *Abstract*

Optical character recognition is an attractive subject for research work now days. In our study of this topic for Bangla Character domain, we have come around some methods of feature extraction and classification along with various preprocessing steps. We have exploited these techniques for their advantages and disadvantages. Among these, the methods that came to our attention because of their high accuracy are Stroke Extraction and encoding which is mainly mathematical, Chaincode extraction based on the direction of the points and Principal Component Analysis(PCA) of the character image. In our research, we have combined these methods and implemented the system in Matlab that gave us better accuracy than their individual implementation.

# Table of Contents

# 1 Introduction

Optical character recognition (OCR) is the automatic conversion of scanned or photographed images of typewritten or printed text into computer readable text.

The concept of OCR is quite old. It has been around in one form or another for about 200 years. But only in the 1950s when the computer was in use, the concept of OCR was realized properly. After that OCR was used as a means for data entry.

The general use of OCR is as a form of data entry from some sort of original paper data source, whether passport documents, invoices, bank statement, receipts, business card, mail, or any number of printed records. It is a common method of digitizing printed texts so that they can be electronically edited, searched, stored more compactly, displayed on-line, and used in machine processes such as machine translation, text-to-speech, key data extraction and text mining. OCR is a field of research in pattern recognition, artificial intelligence and computer vision.

There are actually two distinctly different types of OCR; Offline (software based) and Online (machine based). While the core algorithms share a lot, the similarity stops there. Both technologies are used on very different types of text and have very different ways to tune.

**Online OCR**: For online OCR, it is done at scan time, and very often not on documents rather on objects going down an assembly line. Automatic conversion of text as it is written on Special digitizer, PDA, Mobile devices or any Touch sensitive surface. It is dynamic, real timed and pressure sensor based. Digital Ink concept is widely used in sector. Sensor picks up the pen-tip movements as well as pen-up/pen-down switching. Online OCR for documents is used primarily for mail-room processing on high speed high volume scanners, or on manufacturing assembly lines. Both scenarios need data from the input asset quickly. The benefit of online OCR is it's the fastest OCR around. Usually the OCR is a part of firmware, and optimized for speed. The downside to in-line OCR is accuracy. When documents are scanned the accuracy cannot compare to that of PC based OCR.

**Offline OCR:** Offline means the text written on the plain paper or sheet and then the writing is usually captured optically by a scanner and the completed writing is available as an image. It is an automatic conversion of text in an image into letter codes. Document is scanned as image and Produces output in a character by character format. Output can be processed further as normal word document. It is static and image processing tools and methods are widely used in this field.

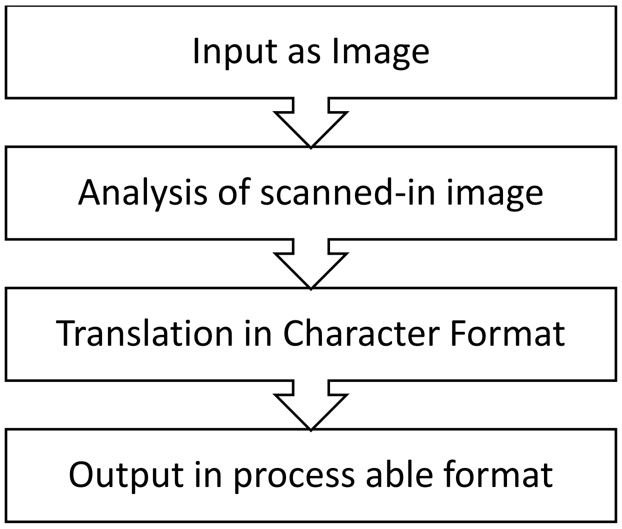

Input as Image
Analysis of scanned-in image
Translation in Character Format
Output in process able format

Fig: General steps of OCR

## 1.1 Problem Domain

The basic hurdle in any form processing system is in recognizing the handwritten characters contained in the form. Several works have been done on the recognition of English handwritten characters. Software is also available to process forms filled up in English language. But no significant work or software is noticed till date to process forms filled up in Bengali (also called 'Bangla') language, which is the one of most spoken-and-written language in Bangladesh.

## 1.2 Properties of Bangla Characters

The Bangla script is derived from the ancient Brahmi script through various transformations. There are 11 vowel and 39 consonant characters in modern Bangla alphabet. They are called basic characters. For many characters there exists a horizontal line at the upper part. It is called the head line. The head line is an important feature to locate the script line, to segment the characters in a word and to classify the characters. A consonant following (proceeding) a consonant is represented by a modifier called consonant modifier. Modifiers are those symbols which do not disturb the shape of the basic characters (in middle zone) to which they are attached. If the shape is disturbed in the middle zone, the resultant shape is called compound character shape. Compounding of two constants is most abundant although three consonants can also be compounded. There are about 250 compound characters of which a sub-set of 100 characters. The total number of basic, modified and compound characters is about 300. A word may be partitioned into three zones. The upper zone denotes the portion above the head line, the middle zone covers the portion of basic (and compound) characters below head line and the lower zone is the portion where some of the modifiers can reside. The imaginary line separating middle and lower zone is called the base line. A typical zoning is shown in the following figure.

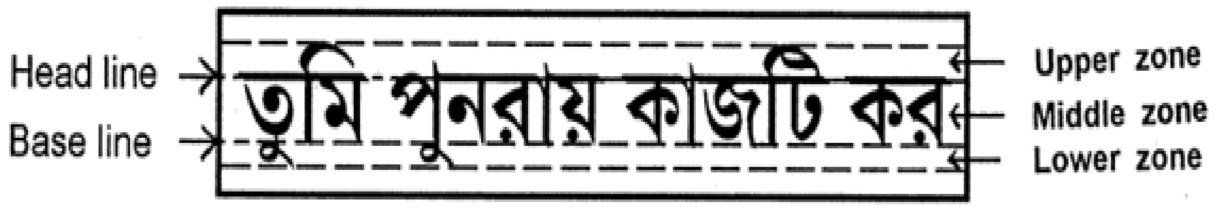


Fig: Zones in charaters

## 1.3 ResearchChallenges

The challenge in handwritten character recognition is mainly caused by the large variation of individual writing styles. Hence, robust feature extraction is very important to improve the performance of a handwritten character recognition system. Again, the difficulty with Bengali character recognition is the large number of classes that exist in Bengali. There are 10 numerals, 49 basic characters, and more than 150 compound characters present in this language. Added to this, the presence of vowel modifiers complicates things to a large extent. Some challenges that are normally faced in research of this topic is mentioned below-

- *Overlapping characters:* In handwriting recognition of any language, there are variations in the style of writing. Bangla characters are a bit complex in shape than other languages. So variations in wrting makes the character recognition a bit more complex. Overlapping of characters can often occur in case if fast writing. Two boundary characters of two adjacent words can overlapp that makes the word seperation complex for those two words. Even two adjacent characters can overlap within a word that makes the the character separation for those two characters complex.

- *Overlapping lines:* Skewness and slant in characters, words and lines make it difficult for segmentation. Mostly skewness affects lines. Lines can overlap in between themselves which can be understood barely by human eyes but not for the computer. It makes it difficult to separate the overlapped lines.

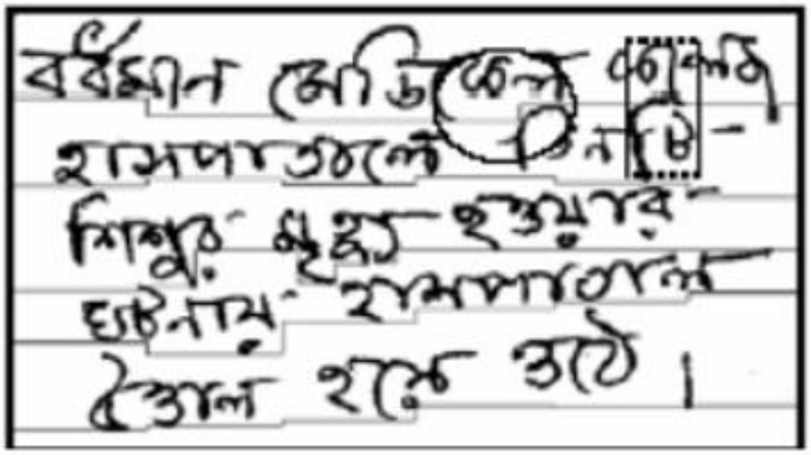


Fig: overlapping of characters and lines

- *Compound characters:* Compound characters are one of the most attractive and difficult to understand feature of Bangla characters. These characters have complex structures which is not fully understandable to any of the existing algorithms of OCR. Even variance in handwriting makes it even more difficult to get higher accuracy in properly identifying the compound characters. So many of the researchers excluded the compound character identification part.

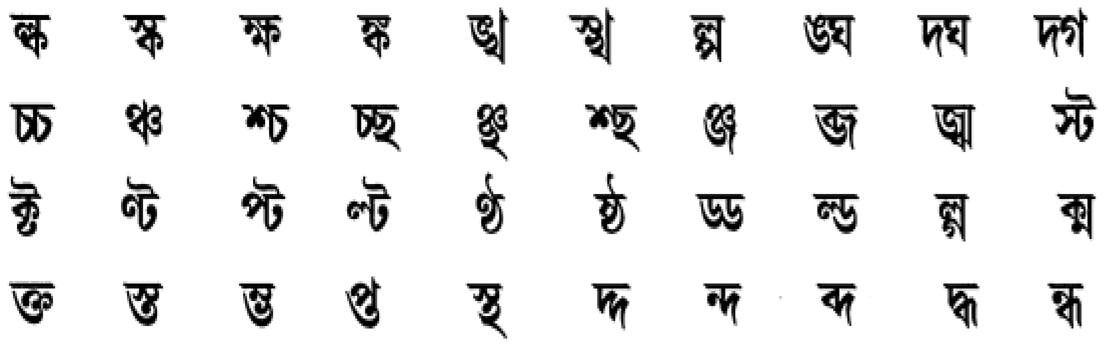


Fig: a subset of compound characters

- *Base line detection:* Among the different zones of lines or characters, head line and base line are the two most important zones. Base line detection is needed to identify many vowel modifiers. Without detecting the base line, the vowel modifiers will be hard to identify as they can be misinterpreted to get mixed with the basic shape of the character with which it is attached to due to different styles in hand writing.

- *Slant in writing:* Slant or italic style writing is one of the main catalyst for the variance in handwriting styles. Characters can be slanted leftwards or rightwards or in both directions in the same word. So slant is an obstruction in proper recognition of the characters.

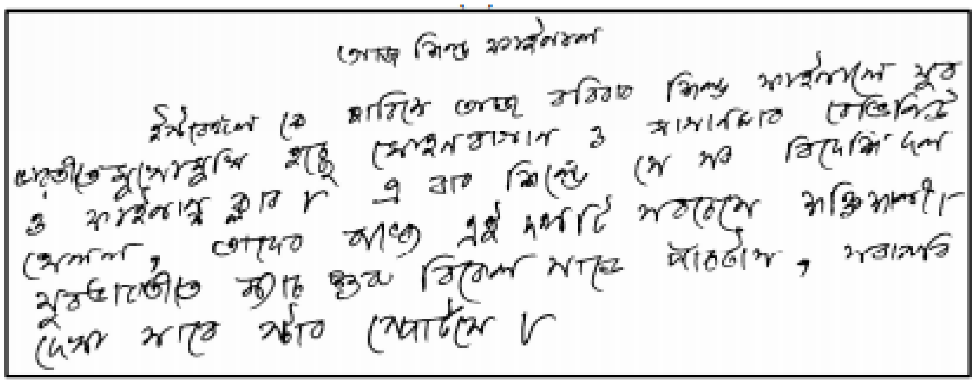


Fig: slant in characters

- *Noise:* Though noise is now a days regarded as a minor challenge due to advanced techniques used in noise removal methods, it can be a major obstacle in proper identification of characters. Due to noise, a character can get mixed with any other characters or the shape can be degraded even upto the point where it cannot be restored to its original shape.

## 1.4 Phases of OCR

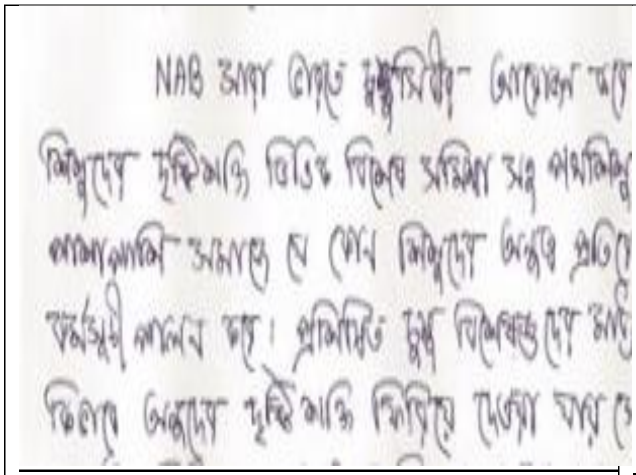
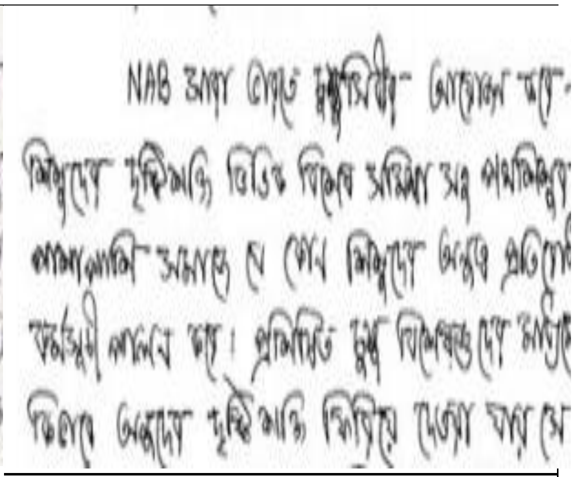
There are some standard procedures in OCR. In every application there is a general form that is followed. There are fundamentally three steps in OCR:

1) Preprocessing
2) Segmentation
3) Feature Extraction
4) Classification

### 1.4.1 Preprocessing

Before we dive into recognition we need to prepare our digital image for recognition. For that some modification to the image is done so it is convenient for us to recognize the part of choice. There are several methods that are applied. But not all of them are mandatory in every case. These steps are described below.

- *Thresholding:* Thresholding is the operation of binarizing the digital image. We set a threshold value and set two different values generally white and black to pixels that cross and don't cross the value respectively. Sometimes we have color images. In that case it needs to be converted to gray scale image before thresholding.

| | |
|---|---|
|  |  |
| Fig: Before Thresholding | Fig: After Thresholding |

- *Slant Correction:* Sometimes the characters in the image are italic or slanted. They can be slanted to the left or slanted to the right. We need although it may not be needed depending on the recognition process to correct the slant angle for the improvement of processing.

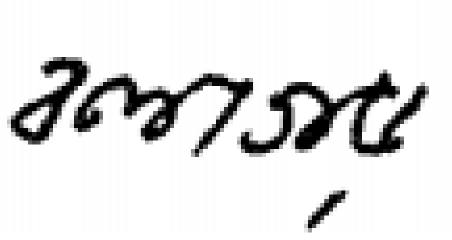


Fig: Slanted Word

- *Skew detection and Correction:* Sometimes the text is skewed. This hinders the recognition process. So we need to detect whether there is any skew and if any correct it.

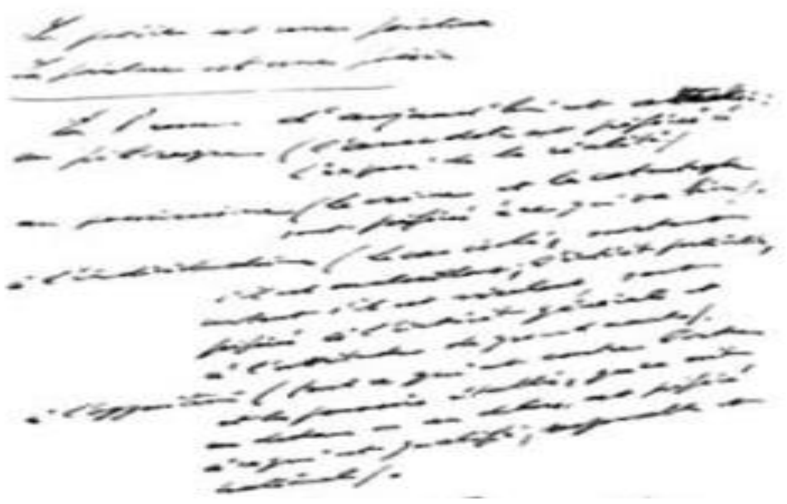


Fig: Skewed Text

- *Noise Removal:* Sometimes there is unwanted information in the scanned image. This is due to dirt on the document or mechanical problem of the scanner or any other input device. So we need to remove it for accurate recognition. A sample of noise is given below.

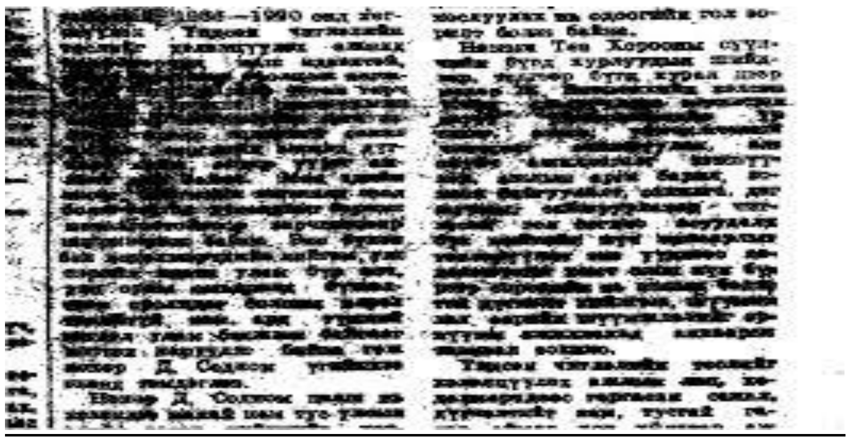


Fig: Noisy Text Image

## 1.4.2 Segmentation

Now that we have prepared our sample image for the task ahead we need to separate each of the components of the text to the letter level. Since it cannot be processed whole, we need to segment it to unit components so that we can apply our algorithm to recognize it. There are three steps in segmentation.

- *Line Segmentation:* A text may have one or more lines in it. They may be straight or curved or irregular. So we need to segment them properly to increase the odds of recognition.

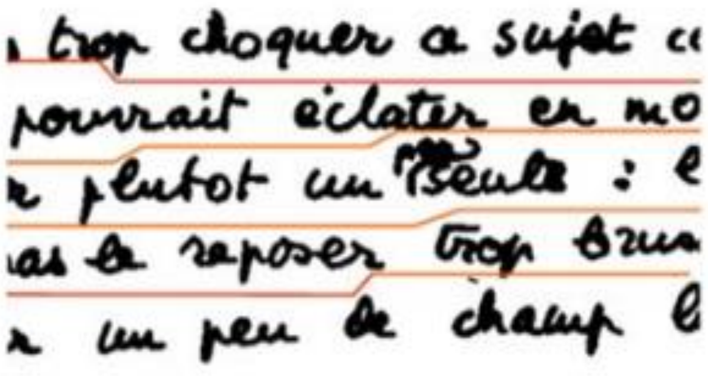


Fig: Line Segmentation

- *Word Segmentation:* After line segmentation comes the word segmentation. It is very vital that every word is segmented accurately. Because after the processing we need to recognize the words and put together. So, if we miss segment the words then some part of the word may be recognized as a part in another word which can be fatal in real life application.

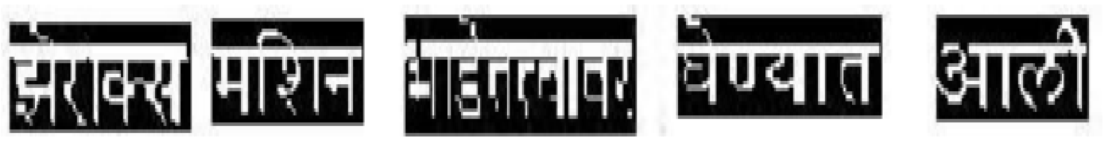


Fig: Segmented word

- *Character Segmentation:* Now comes character segmentation which is more important than the above segmentations. Because the rate of successful identification of a word is mostly dependent on character segmentation. If we cannot segment the character properly then the recognizer may not recognize it at all.

Fig: Before Segmentation

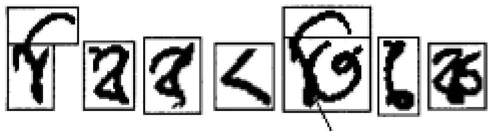


Fig: After Segmentation

### 1.4.3 Feature Extraction

Before we start our recognition process we need to define the properties by which we can recognize a character for what it is. There are many different properties of a character which varies from script to script for different languages. We need find the properties that will give us the best recognition. There are several types of features. Some give us fast recognition than others and some gives us more accurate. Depending on our demand the feature is chosen. These feature gives us a sequence of numbers called feature vector. This feature vector is then used as the input of the classifier which is described next.

### 1.4.4 Classification

This is the recognition step.Here the feature vector is given to the classifier and it outputs the best match. Sometimes there is a fixed database. The classifier matches the input feature vector with the feature vector of the database. Then it outputs the best match. In another case, the classifier can be taught to identify certain types of handwriting. In this case a learning step is included for the classifier. This enables the classifier to learn the input character features and add it to the database.

## 2 Related Work

Many features have been developed in the last few decades. M. Shi et al. studied the use of gradient and curvature of gray-scale character images to improve the recognition accuracy. They presented three procedures to estimate the curvature of gray-scale curves, based on curvature coefficient, bi-quadratic interpolation, and gradient-vector interpolation. They composed a feature vector of the gradient and the curvature by simple concatenation and cross product. Results show that the direction of gradient is necessary for shape discrimination and the composite features by the cross product to achieve a higher recognition rate. S.-W. Lee utilized the coefficients of wavelet transform as a feature for character recognition. They suggested that character images of different resolutions characterize different structures of the character. This method actually denotes a global feature in multi-resolution analysis. G. A. Fink et al. have proposed an online Bangla handwriting recognition that considers cursively written words instead of isolated characters. It uses a sub-stroke level feature representation of the script and a writing model based on hidden Markov models. Mane and Ragha have proposed an elastic image matching technique for recognition of offline isolated English handwritten digits by matching against a sequence of templates. During pre-processing, it reduces some undesirable variability by filtering, normalization, segmentation, etc.; template matching is based on Euclidean and Mahalanobis similarity measures.

## 3 Contribution

In our research, we have used three different feature extraction methods namely stroke detection and scoring, Chaincode and PCA. We have taken these three mechanisms and generated three different feature vectors. These three feature vectors signify the structural and holistic features and principal components of the characters. We have taken these three different types of features and combined them to be a single feature that will help to get a better classification result. For classification we have used multi class SVM which is a modified version of the two class SVM classifier that is available in Matlab. Besides, in the preprocessing part of our implementation we have developed some algorithm designed for handwritten character segmentation better suited for our problem domain and specifically our proposed feature extraction method.

# 4 Existing Work

## 4.1 Projection Histogram

In projection histogram the horizontal histogram is computed of the input image. Then a line is drawn in the row position where the projection value is lowest that is zero and between two high value projections. It means we will draw a line between two lines and separate them accordingly. This method is good for text with significant gap between lines. But it is not suitable for touching lines. In that case Arivazhagan et al.proposed a piece wise projection histogram. In this method the input image is divided into a number of columns. Then the horizontal histogram of each column is taken. This will give a figure like below. Then the valleys of the histogram of each column is added with a line. If there is no valley but a peak then the line goes around the peak and adds the peak to the top line or bottom line according to the joint probability of the already connected component using the chain rule of probability.

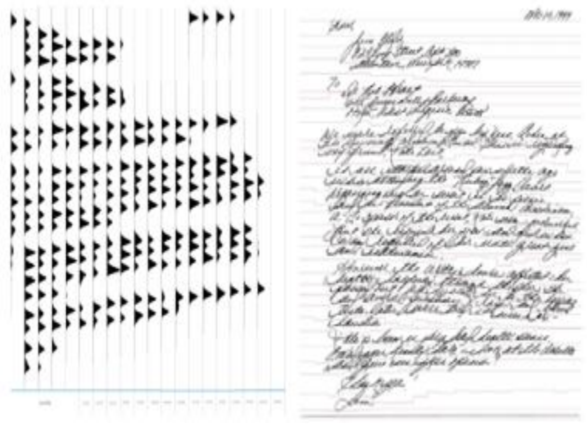
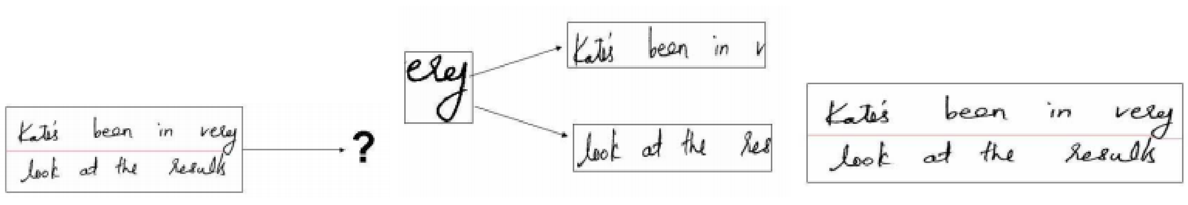


Fig: Piece-wise Histogram Projection

$$P(C|\vec{\mu}_A, \Sigma_A) = P(p_1|\vec{\mu}_A, \Sigma_A) \cdot P(p_2|\vec{\mu}_A, \Sigma_A, p_1) \dots P(p_T|\vec{\mu}_A, \Sigma_A, p_1, p_2, \dots, p_{T-1})$$

Fig: Chain rule of Joint Probability



(a) A line drawn is obstructed by the descender of the letter 'y'.

(b) A decision need to be made whether to associate this connected component to the above or the below line.

(c) A line traversing below after deciding that the component belongs to the line above

**Advantage:** This method is robust to scripts with skew and lines running into each other. This method does not require the input to be skew corrected. So, we can avoid this portion in the preprocessing reducing calculation complexity.

## 4.2 Smearing Approach

This approach was proposed by **Li et al.** In this technique, consecutive black pixels along the horizontal direction are smeared. If the distance between the white space is within a predefined threshold, it is filled with black pixels. The bounding boxes of the connected components in the smeared image are considered as text lines. They first convert a binary image to gray scale using a Gaussian window, which enhances text line structures. Text lines are extracted by evolving an initial estimate using the level set method. Preliminary experiments show that their method is more robust compared to a bottom-up connected component based approach. Examples show that the method is script independent. This has been qualitatively confirmed by testing it on handwritten documents in different languages, such as Arabic, English, Chinese, Hindi, and Korean.

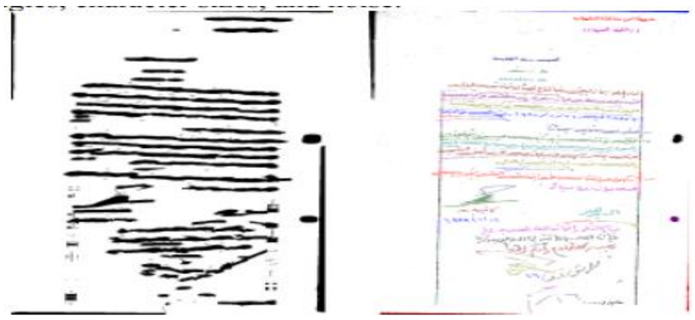


Fig: Smearing Approach

## 4.3 Graph Based Approach

Graph cut is one of the methods used to perform image segmentation. It promises a near optimal solution; i.e., a solution at a known distance from the global optimum. To apply graph cuts to document images, a graph is built using either the pixels or the connected components of the image as nodes, which are linked to its neighbors through edges. During segmentation, a cut is defined on the graph, which labels the pixels or components on either side of the cut as belonging to different segments. Boykov et al. proved that minimizing an energy function is equivalent to minimizing the cost of cut on the graph. Our goal is to assign a line number (label) to every connected component within a document. All the connected components that belong to a document need to be partitioned into mutually exclusive and collective exhaustive subsets based on line. The goal is to find a labeling that labels the connected component in a line. The labeling should be done in such a way that it is piecewise smooth. In this framework, a labeling f is computed so as to minimize the total energy:

$$E(f) = E_{smooth}(f) + E_{data}(f)$$

Here f is the label set. A labeling is extremely **smooth** if its K-nearest neighbors fall in a single line. The **Data** term uses priori information like line number to decide whether a component should be connected with the upper line or lower line.

$$E_{smooth}(f) = \sum_{(c,c')\in\mathcal{N}} V_{(c,c')}(c,c') \qquad E_{data}(f) = \sum_{c\in\mathcal{C}} D(f(c))$$

Here ***c* and *c'*** denotes a component.

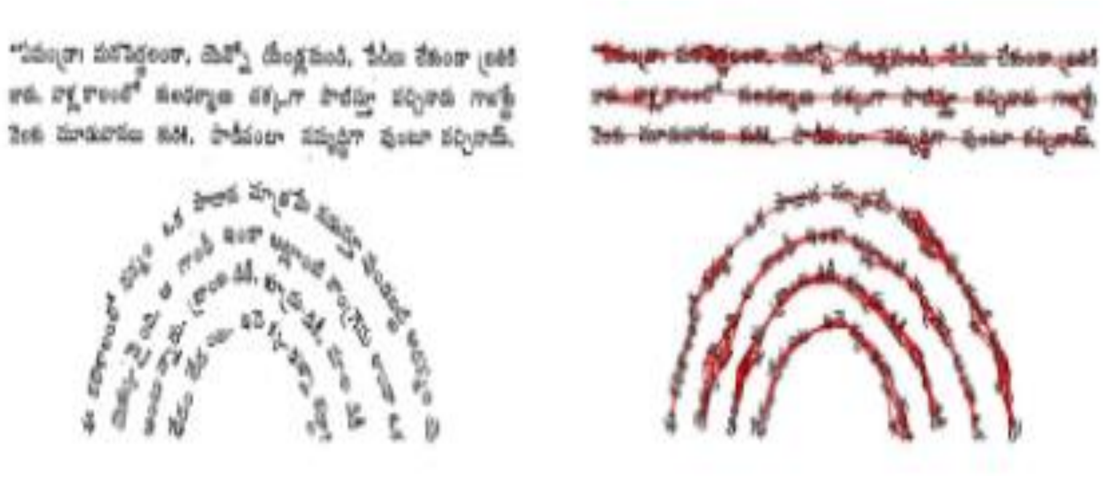


Fig: Graph Cut Method

## 4.4 CTM Approach

The CTM method finds a path or cut line in between the text lines to be separated which minimizes the text line pixels cut by the segmentation line, especially descenders from the upper line and ascenders from the lower line. The method attempts to track around ascenders or descenders to avoid cutting them. If the deviation is too great, the segmenter aborts and continues its forward path. A rough estimate of text line separations were first obtained using vertical projection histograms.

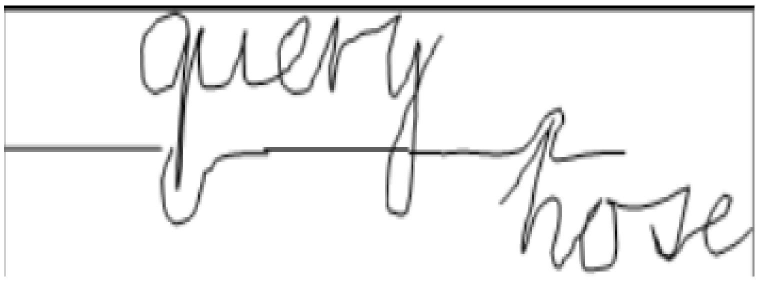


Fig: Avoiding ascenders and descenders

## 4.5 Vertical Histogram

After a text line is segmented, it is scanned vertically. The column wise scan ensures detection of the gaps among words in a line. The process is repeated for each line in a document. If in one vertical scan two or less black pixels are encountered then the scan is denoted by 0, else the scan is denoted by the number of black pixels. In this way a vertical projection profile is constructed. Then if there is a gap between two projections that crosses certain threshold the characters projecting them are separated as a word.

**Advantage:** If the words have significant gaps among them, then this method gives high performance with accuracy. But the threshold must me set accordingly or with convention for proper segmentation.

**Limitation:** For overlapping word, this method can’t give proper segmentation as there is no gap in between two boundary characters of two adjacent words.

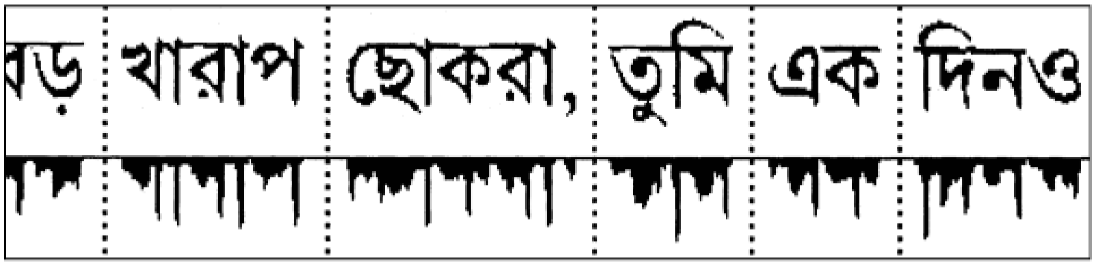


Fig: Vertical histogram

## 4.6 Piece-wise Linear Scanning

This is a character segmentation process highly robust for basic Bangla characters. To find the characters a linear scanning in the vertical direction from the head line is initiated. If during a scan, one can reach the base line without touching any black pixel then this scan marks a boundary between two characters. Otherwise it proceeds with the 8-neighboring pixels of the black pixel found on that path and again tries to reach the base line from that neighbor. This process follows for every black pixel found on the way until it reaches the base line.

**Advantage:** this method is very effective if there is no vertical gap between them or the characters have non-vertical shape contents in vertical orientation.

**Limitation:** one of its limitations is it can't segment for overlapping character. In fact it gives a calculation overhead if it can't reach the base line. So a limit is given to indicate how many neighbors it have to check to ensure that the starting scan point is a part of a character that initially starts from another vertical scan point.

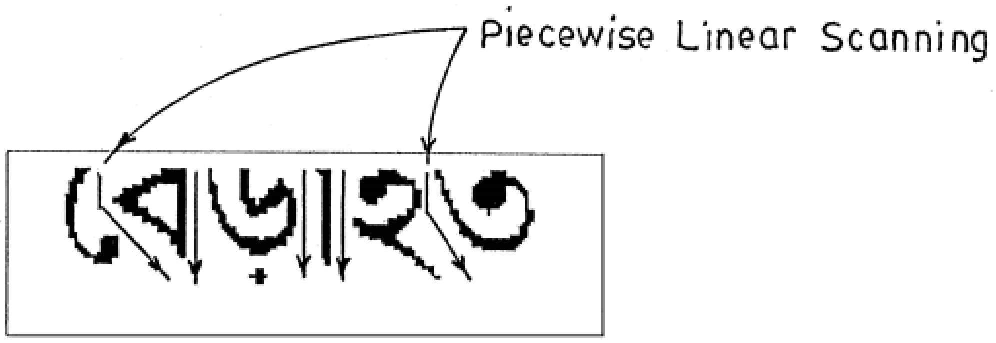


Fig: Piece-wise linear scan

## 4.7 Water Reservoir

The water reservoir principle is as follows. If water is poured from a side of a component, the cavity regions of the background portion of the component where water will be stored are considered as reservoirs of the component. These reservoirs are used for the segmentation of the words into primitives. Some of the water reservoir principle based features are:

**Top (Bottom) reservoir:** By top (bottom) reservoirs of a component we mean the reservoirs obtained when water is poured from top (bottom) of the component. A bottom reservoir of a component is visualized as a top reservoir when water will be poured from top after rotating the component by 180°.

**Water reservoir area**: By area of a reservoir we mean the area of the cavity region where water will be stored. The number of pixels inside a reservoir is computed and this number is considered as the area of the reservoir.

**Water flow level:** The level from which water overflows from a reservoir is called as water flow level of the reservoir.

**Height of a reservoir**: By height of a reservoir we mean the depth of water in the reservoir.

**Base-line**: A line, passing through the deepest point of a reservoir and parallel to its water flow level is the base line

**Advantage:** The reservoirs can identify those characters that have hollow area in the bottom part. It gives high accuracy for these characters. For this aspect, it is more robust than piecewise linear approach.

**Limitation:** It cannot identify compound characters due to variation in handwriting style from user to user.

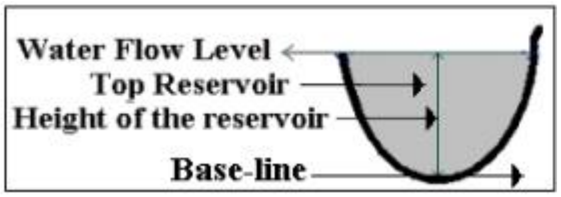


Fig: Water Reservoir features

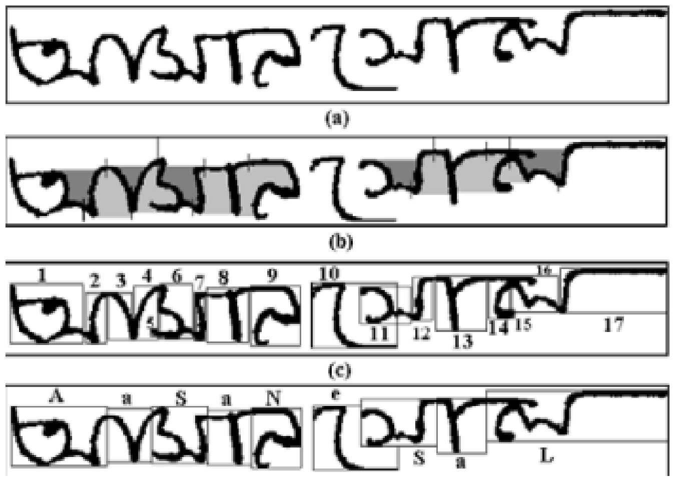


Fig: Water reservoir principle

## 4.8 Template Matching

In binary template matching, several similarity measures other than mean square distance and correlation have been suggested. To detect matches, let i, j be the number of pixel positions where the template pixel x is i and the image pixel y is j, with i, j Є {0, 1}

$$n_{ij} = \sum_{m=1}^{n} \delta_m(i, j)$$

where

$$\delta_m(i, j) = \begin{cases} 1 & \text{if } (x_m = i) \wedge (y_m = j) \\ 0 & \text{otherwise,} \end{cases}$$

$\boldsymbol{y_m}$ and $\boldsymbol{x_m}$ are the m-th pixels of the binary images Y and X which are being compared. This method lacks robustness in case of shape variation. This was reduced by introducing weights to the different pixel positions.

$$n_{ij} = \sum_{m=1}^{n} p_m(k|i)\delta_m(i, j)$$

But this method cannot cope with variability in character shape.

## 4.9 Zoning

In this method character image is divided in different zones. There are many approaches for zoning. Such as an *nxm* grid is superimposed on the character image, and for each of the *nxm* zones, the average gray level is computed giving a feature vector of length *nxm.* Some use center of gravity for zoning whereas some use density of pixels for zoning dividation.

**Advantage:** The dividation into zoning and then calculating the respected values for the zones is a very fast process to implement.
**Limitation:** if the selection of criterion for zoning is miscalculated then it can lead to failure of detection and misinterpretation of the characters.

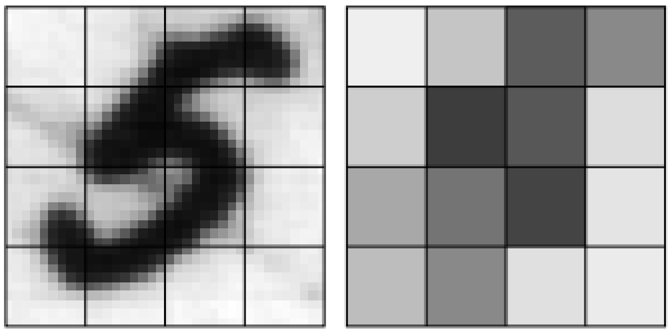


Fig: Zoning

## 4.10 Zernike Moment

Zernike moment is well suited for gray-scale character sub images as well as binary images. Both rotation-variant and rotation-invariant features can be extracted. Features invariant to illumination need to be developed for these features to be really useful for gray level character images. Khotanzad and Hong used the amplitudes of the Zernike moments as features. A set of complex orthogonal polynomial $\boldsymbol{V_{nm}(x, y)}$ is used. The Zernike moments are projections of the input image onto the space spanned by the orthogonal $\boldsymbol{V}$-functions.

$$V_{nm}(x, y) = R_{nm}(x, y)e^{jm\tan^{-1}\frac{y}{x}}$$

where $j = \sqrt{-1}$, $n \geq 0$ , $|m| \leq n$ , $n - |m|$ is even, and

$$R_{nm}(x, y) = \sum_{s=0}^{\frac{n-|m|}{2}} \frac{(-1)^s (x^2 + y^2)^{\frac{n}{2}-s} (n-s)!}{s!\left(\frac{n+|m|}{2} - s\right)!\left(\frac{n-|m|}{2} - s\right)!}$$

For a digital image, the Zernike moment of order $n$ and repetition $m$ is given by

$$A_{nm} = \frac{n+1}{\pi} \sum_x \sum_y f(x, y)[V_{nm}(x, y)]^*$$

Then the Zernike moment of the image is calculated with the following equation.

$$f(x, y) = \lim_{N \to \infty} \sum_{n=0}^{N} \sum_m A_{nm} V_{nm}(x, y)$$

## 4.11 Chain Code

A chain code is a lossless compression algorithm for monochrome images. The basic principle of chain codes is to separately encode each connected component, or "blob", in the image. For each such region, a point on the boundary is selected and its coordinates are transmitted. The encoder then moves along the boundary of the region and, at each step, transmits a symbol representing the direction of this movement. This continues until the encoder returns to the starting position, at which point the blob has been completely described, and encoding continues with the next blob in the image. For handwriting recognition, an 8-direction Freeman Chain Code is used to represent the time-series data of the stroke. The Freeman code carries connectivity and geometric information. Skeletal representation of features in the raster model can be expressed by the Freeman code. This code follows the contour in counter clockwise manner and keeps track of the directions as we go from one contour pixel to the next.

**Advantage:** This encoding method is particularly effective for images consisting of a reasonably small number of large connected components. Good for compound character recognition.

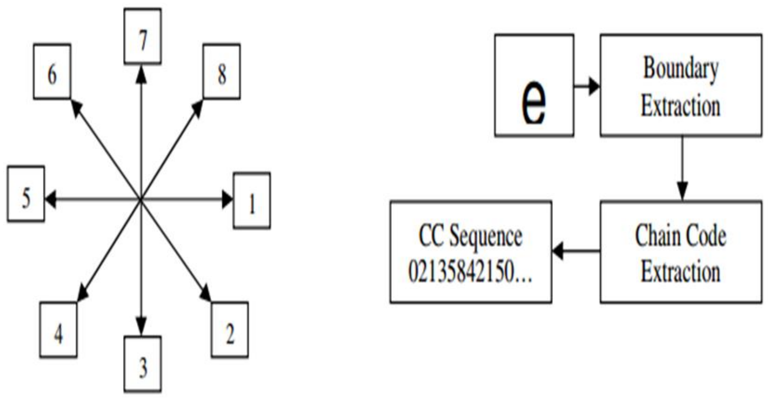


Fig: chain coding

## 4.12 Wavelet Transformation

In this method first the gradient of the image characters are calculated. This is a feature that will be combined with the wavelet transformation to create the total feature vector. The equation to calculate the gradient magnitude and direction is given below.

$$G(i,j) = \sqrt{g_v^2(i,j) + g_h^2(i,j)}, \ \theta = \arctan \frac{g_v(i,j)}{g_h(i,j)}$$

Fig: Gradient feature

Wavelet transform can be regarded as a transformation that maps a signal to the multi-resolution representation. The coefficients of wavelet transform for a character image give us a scale-invariant representation in multi-resolution analysis. It decomposes a function by a set of basic function called wavelets. The function for wavelets can be described like below:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right)$$

And we can define the wavelet transformation like:

$$W(a,b) = \int_t f(t)\frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right)dt$$

Here ***a*** represent the scalar factor and ***b*** represent the translation factor. W is called the wavelet transformation coefficient representing how much the scaled wavelet is similar to the function ***f(t) =b/a.*** Then we combine the wavelet coefficient with the gradient features to create the whole feature.

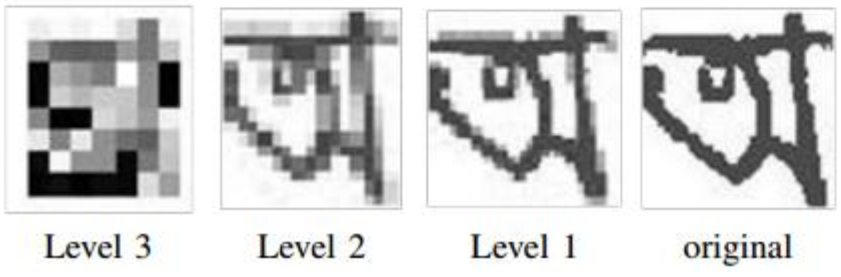


Fig: Wavelet transformation up to level 3

## 4.13 Multilayer Perceptron

An MLP is a feed-forward layered network of artificial neurons. Each artificial neuron in the MLP computes a sigmoid function of the weighted sum of all its inputs. An MLP consists of one input layer, one output layer and a number of hidden or intermediate layers, as shown in Fig. The output from every neuron in a layer of the MLP is connected to all inputs of each neuron in the immediate next layer of the same. Neurons in the input layer of the MLP are all basically dummy neurons as they are used simply to pass on the input to the next layer just by computing an identity function each. The numbers of neurons in the input and the output layers of an MLP are chosen depending on the problem to be solved. The number of neurons in other layers and the number of layers in the MLP are all determined by a trial and error method at the time of its training. An ANN requires training to learn an unknown input-output relationship to solve a problem.

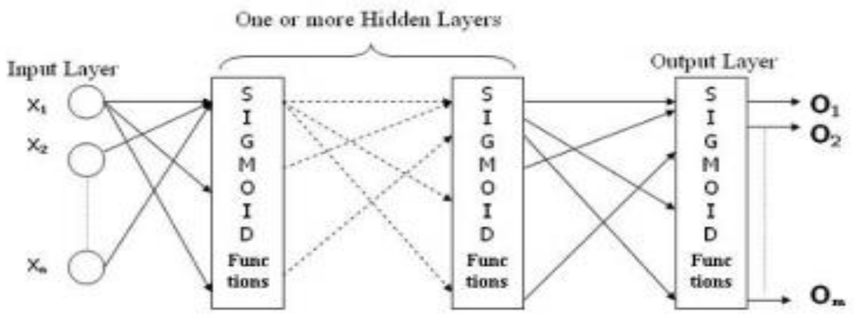


Fig: Multilayer Perceptron

## 4.14 K-NN

The *k*-Nearest Neighbors algorithm is a non-parametric method used or classification. The input consists of the *k* closest training examples in the future space. In *k-NN classification*, the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors. If *k* = 1, then the object is simply assigned to the class of that single nearest neighbor. *K*-NN is a type of instance-based learning, where the function is only approximated locally and all computation is deferred until classification. The *k*-NN algorithm is among the simplest of all machine learning algorithms. The neighbors are taken from a set of objects for which the class is known. This can be thought of as the training set for the algorithm, though no explicit training step is required. In handwriting character recognition k-NN is a good approach for complex structured character recognition.

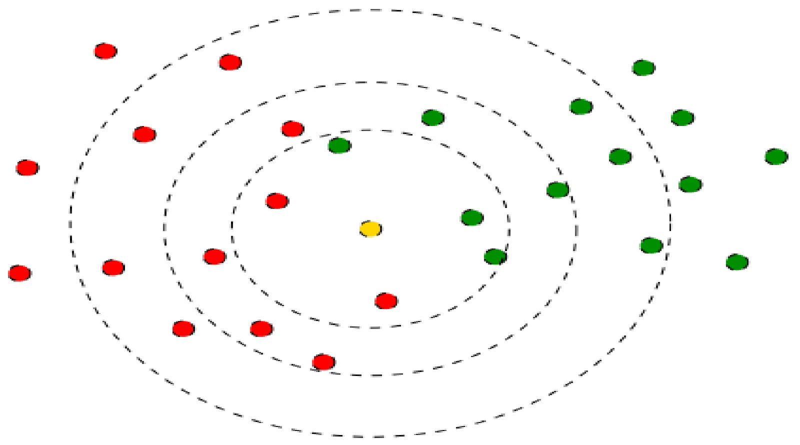


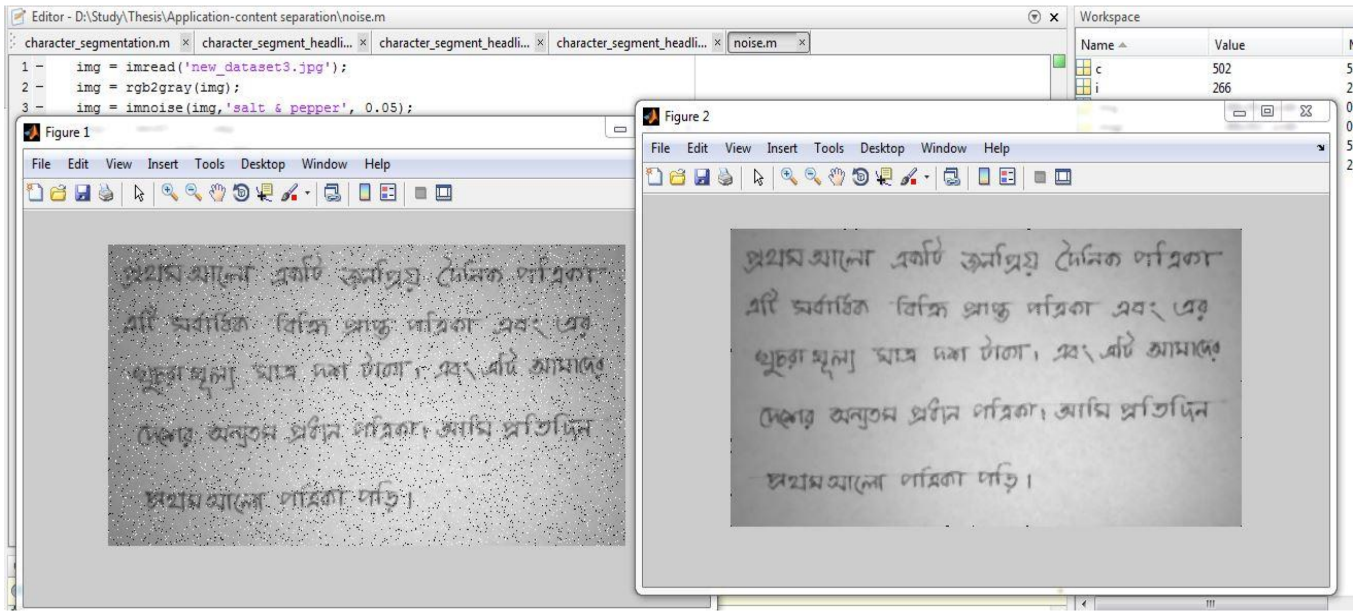
Fig: k-Nearest Neighbor

# 5 Proposed Method

We have a proposed method for our research topic. We are trying to improvise the existing algorithms for the different phases of the optical character recognition so that we get a better result than that of those papers without being identical. Uniqueness is one of our prime priority for such cases. Our proposed method is described below with respect to the different sections of the steps of optical character recognition.

## 5.1 Noise Removal

The noise that is dominant in OCR are Salt and Paper noise. We have removed the Salt and Paper noise with median filter. Even after that noise remains in the image as we know we cannot entirely remove noise without blurring the image. But we need the image as intact as possible. So we have found a threshold suitable for us that balances our need for noise removal and blurring. After that we use morphological operation to remove the remaining noise to a greater extent.


Editor - D:\Study\Thesis\Application-content separation\noise.m

```
img = imread('new_dataset3.jpg');
img = rgb2gray(img);
img = imnoise(img,'salt & pepper', 0.05);
```

Fig: Document with noise (left) and after removing noise (right)

## 5.2 Skewness correction

For skewness detection, we have used the approach of first detecting the two top corner points, then calculating the slope of the line joining those two points, then detecting the bottom two corner points, then calculating the slope of the line joining those two points and at last taking the average slope of the two lines. We then rotated the image in the opposite angle of that slope.

Algorithm:

1. input an image of Bangla handwritten document
2. find the top two corner points

3. take the slope of the lines joining the above two points
4. find the bottom two corner points
5. take the slope of the lines joining the above two points
6. take the average of the two slopes
7. rotate the image in the opposite direction of the slope

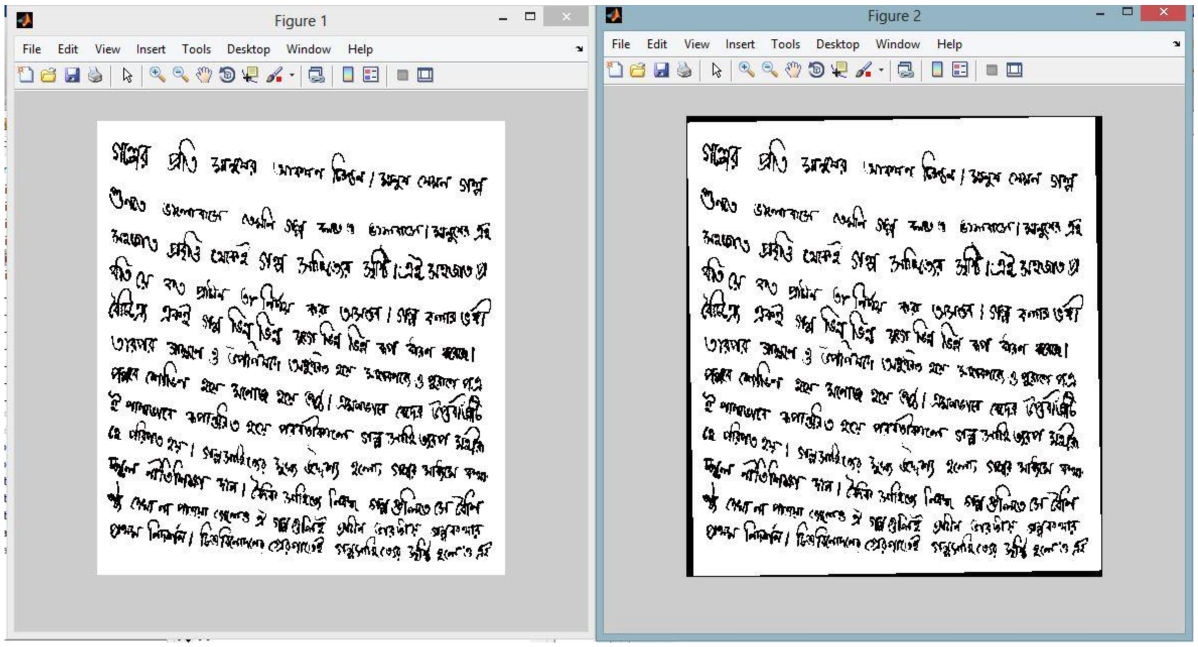


Figure 1
File Edit View Insert Tools Desktop Window Help
Figure 2
File Edit View Insert Tools Desktop Window Help

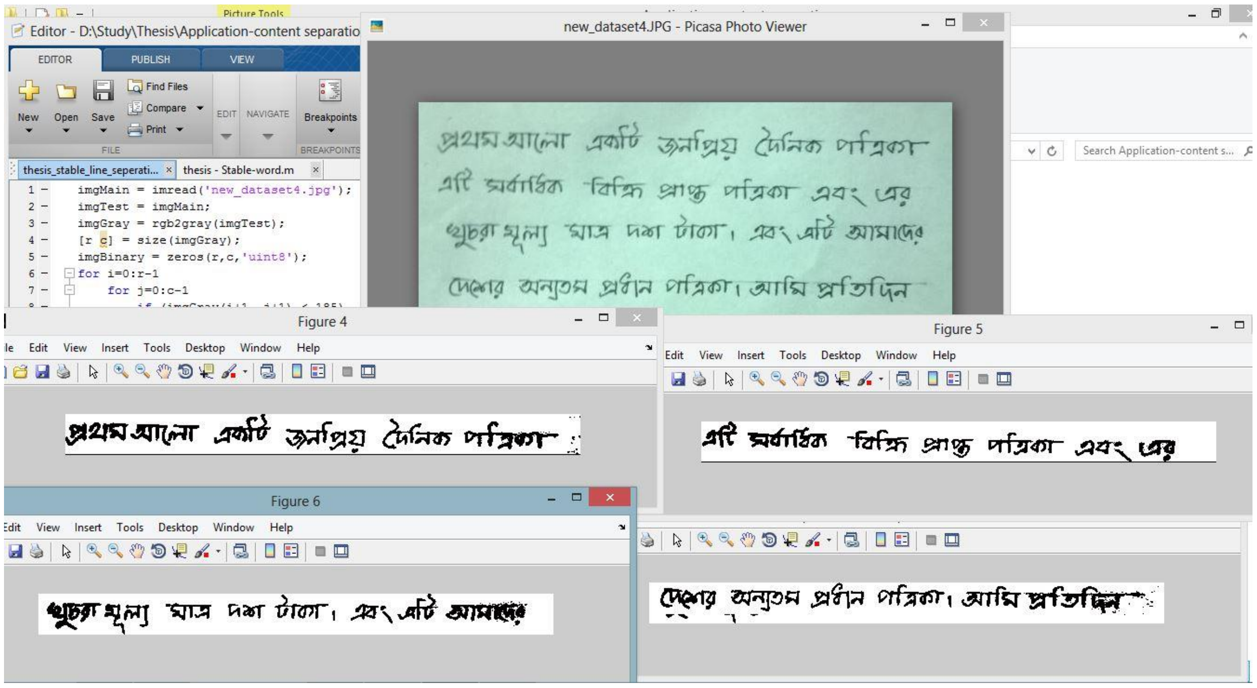
Fig: skewness correction

## 5.3 Line Segmentation

For line segmentation we are using the horizontal histogram approach. We are taking input of the skew corrected image of a handwritten Bangla document through the scanner. First we are omitting the white blank spaces around the rectangle part of the document where there are only texts in it. Then we are taking horizontal histogram in search of only the lines. The lines will give us non-zero data for the histogram analysis of the rectangle part of document with only texts. The in-between spaces of the lines show zero values for those blank spaces. So we get bands of lines and band of white spaces or the in between spaces of the lines. Now we only take the lines which are now separated from the document. But for a test case where multiple lines are overlapped, we didn’t get the result with two separated lines. This approach gave us output of a single line for those two lines as there were no gap between them.

Algorithm:

1. input an skew corrected image of Bangla handwritten document
2. take only the rectangle part of the image with only the lines in it
3. take horizontal histogram of the rectangle part
4. take the bands of histogram with non-zero values which denote the lines of document



```
imgMain = imread('new_dataset4.jpg');
imgTest = imgMain;
imgGray = rgb2gray(imgTest);
[r c] = size(imgGray);
imgBinary = zeros(r,c,'uint8');
for i=0:r-1
    for j=0:c-1
```

Fig: line segmentation

## 5.4 Word Segmentation

For word segmentation we used the vertical histogram approach. After the lines are separated, we ran a vertical histogram algorithm on each of the lines. The words gave non-zero values and in between space of the words gave zero values for the vertical histogram. So we now only take the band with non-zero values that denotes the word. We got better result for test cases in which the word had significant gap between adjacent words. But for words with no gaps with adjacent words, this approach gave a single word for the two words. Also with a single vertical straight line gap between two characters in single word, this approach gave us two separate words instead of a single word. This happened because we set the threshold value as a single pixel gap for two adjacent words.

Algorithm:

1. input the segmented lines
2. take vertical histogram of the lines
3. take the bands of histogram with non-zero values which denote the words of document

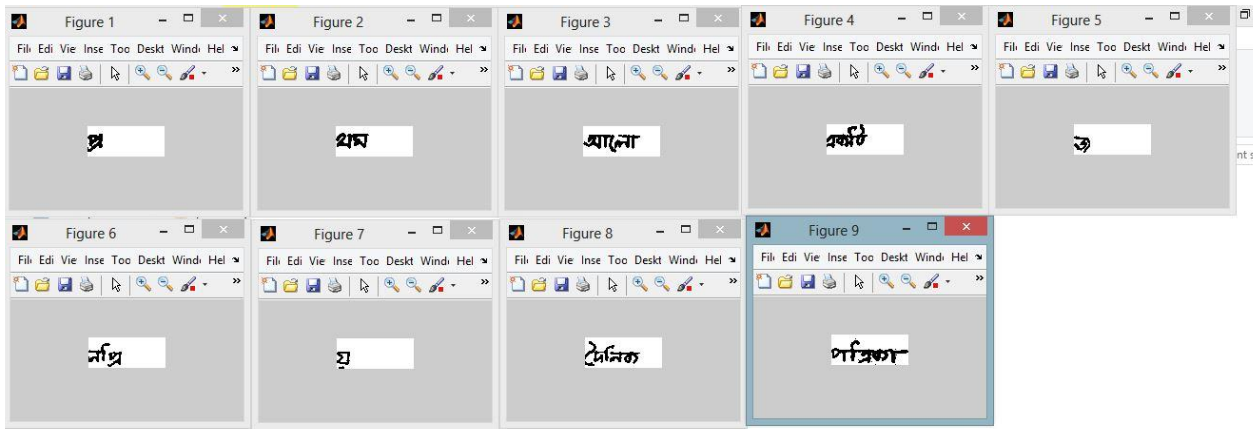


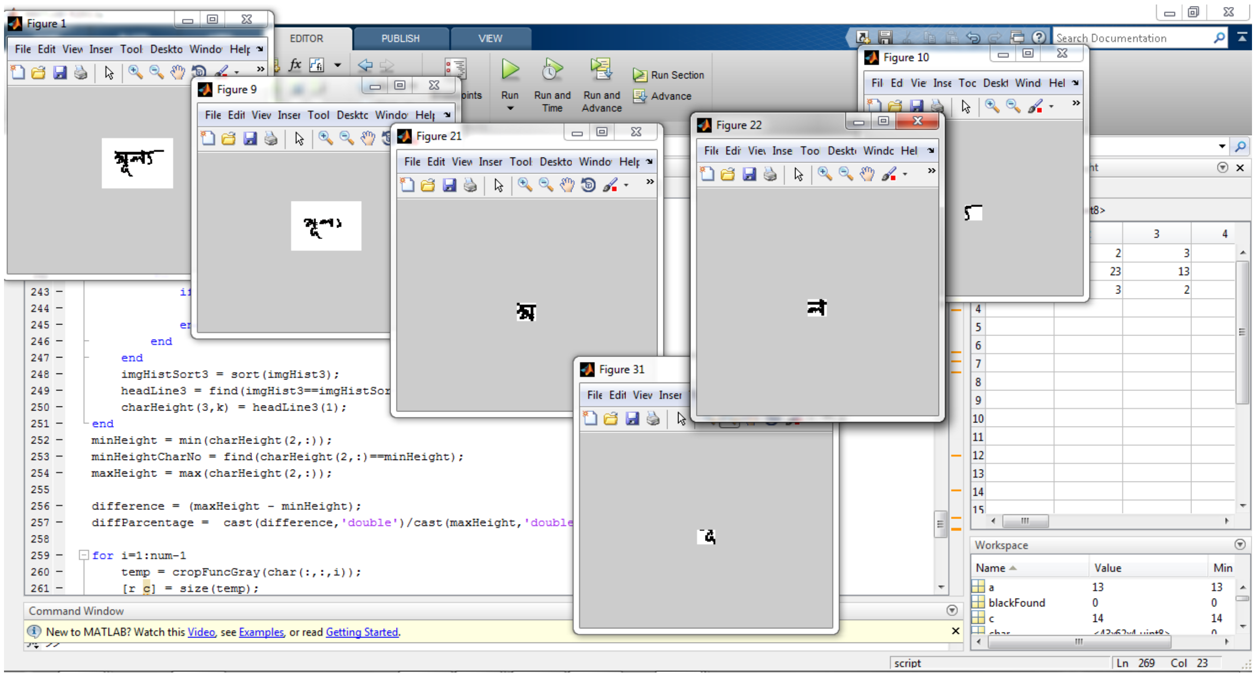
Fig: word segmentation

## 5.5 Character Segmentation

For character segmentation we used the approach of cutting through the white spaces where there is no two adjacent characters are overlapping. First we corrected the skew of the word that we are working with. After detecting the headline of the word, we detect the bands of white spaces where there is no two adjacent alphabet overlaps. The white bands have heights with more than half of the height of the words. This ensures that we don't take any redundant and unnecessary white space bands that occurred due to the curvature of the shape of the character. Now for each of the bands we take each pixel of the headline as a starting point to cut through the white space in a zigzag manner. We move downwards first from the headline and if we get a black pixel at any point in that path and try to move sidewise and then downwards. If at any path we cannot move any further downwards and the path is blocked from all sides then that path is discarded or the starting point from the headline is discarded and we take into account the next starting point of the headline from that band we are working with. The path for which we can reach the bottom of the word is taken as the cutting line. If for any band, we cannot get a perfect path that band is discarded and taken to be only a gap within the characters due to the shape of that character. Now we have all the paths to cut through the words. After cutting through according to those paths we get the bottom part of the headlines. We used this same algorithm to cut through the top of the headline with a slight modification providing the chance to move backwards to choose a different path from the same starting point. The starting points are same as those we get the prefect cutting path for the bottom part of the headline.

<u>Algorithm:</u>

```
1.  Correct the skew of the word
2.  detect the headline of the word using vertical histogram
3.  take bands of whitespaces where the height is more than half of the height of the word
4.  for each band
5.      for each pixel on the headline as the starting point
6.          if found a black pixel as an obstacle
7.              move sidewise
8.              if cannot move sidewise
9.                  then discard this starting point and continue with the next starting point
10.             end if
11.         else
12.             go downwards along the straight line path until reached the bottom
13.         end if
14.         if reached bottom of the word
15.             then note the starting point and the path
16.         end if
17.     end for
18. end for
19. for each of starting point with the perfect path towards the bottom
20.     move upwards toward the top of the word as previously
21.     can move downwards now
22. end for
23. now we have the path from top of the word to bottom through the headline starting point
24. cut the word according to the paths and get the characters separated
```



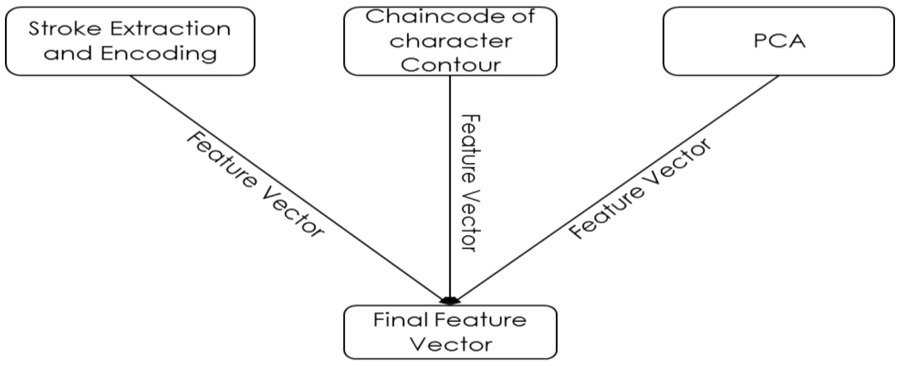
```
            if
            
            er
        end
    end
    imgHistSort3 = sort(imgHist3);
    headLine3 = find(imgHist3==imgHistSor
    charHeight(3,k) = headLine3(1);
end
minHeight = min(charHeight(2,:));
minHeightCharNo = find(charHeight(2,:)==minHeight);
maxHeight = max(charHeight(2,:));

difference = (maxHeight - minHeight);
diffParcentage =  cast(difference,'double')/cast(maxHeight,'double

for i=1:num-1
    temp = cropFuncGray(char(:,:,i));
    [r c] = size(temp);
```

New to MATLAB? Watch this Video, see Examples, or read Getting Started.

Fig: Character Segmentation

## 5.6 Feature Extraction

Our proposed method is based on three different feature extraction methods. They are Syntactic method of Stroke detection and ranking, Chaincode of the boundary of the character and PCA (Principle Component analysis). The strokes of the characters represent the local features of the character. It is related to the structure of the character. This feature gives us a vague description of the construction of the character. The chaincode of the boundary also gives us a local feature that deals with the shape of the character. We use these two features to identify the character by using them together. Then we use PCA to get a reduced feature of the overall image. Then we combine the three features to get a modified feature that we use to identify the character. The overall feature vector acquisition method is given in the figure below:



Now we are going to describe the algorithms and logical factors for these steps in the following section.

### 5.6.1 Image Resize

First we resize the entire character image in a fixed size which is a 50X50 image in our case. We have seen significant increase in the recognition rate if the size of all the character is kept same.

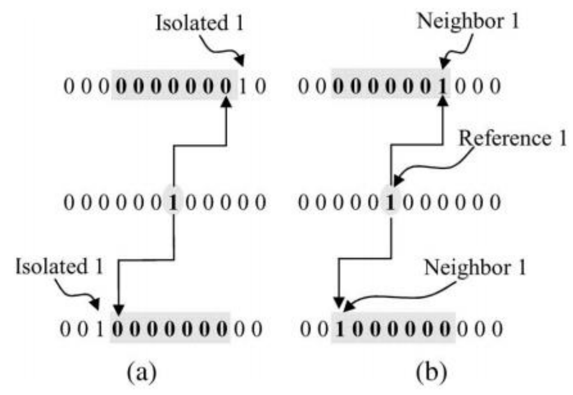
### 5.6.2 Stroke Extraction and Encoding

#### *5.6.2.1 Contour Tracing and filtering*

In this step we take the input image and we replace the first bit of a sequence of consecutive character bits with only one instance of those bits. This is called contour tracing. Then we filter out the contours that are below some threshold so if there are some leftover noise in the image then those will be removed.

#### *5.6.2.2 Stroke Extraction*

Next we extract each and every remaining continuous lines or curves in the filtered image. Points in a line are deemed continuous if three bits on either side in its immediate upper line are black, as depicted in Fig. 8. If a discontinuity occurs, scanning proceeds to the next line with the assumption that the void in the current line is due to improper scanning. If the next line presents continuity, a black point is assumed in the previous line because improper scanning may result in the loss of a pixel, and its coordinates are saved. Otherwise, the stroke is assumed to have terminated.



(a) Isolated "1." (b) Digit "1" having neighbor points.

#### 5.6.2.3 Stroke Encoding

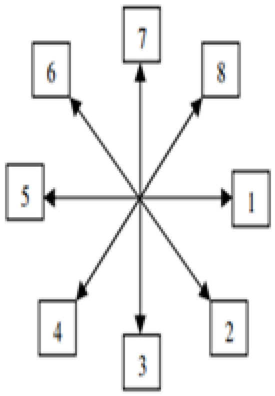
Next we take the saved strokes and identify them using algebraic equation as one of the following types in the table and encode them with the related number.

| Stroke | Characteristics | Numeric Code |
|---|---|---|
| — | Horizontal line | 1 |
| / | Positive sloping line | 2 |
| \| | Vertical Line | 3 |
| \ | Negative sloping | 4 |
| ⊂ | Vertical concave | 5 |
| ⊃ | Vertical convex | 6 |
| None | Rejected stroke | 0 |

This number is then used as a feature vector.

### 5.6.3 Chaincode Extraction

A chain code is a lossless compression algorithm for monochrome images. The basic principle of chain codes is to separately encode each connected component, or "blob", in the image. For each such region, a point on the boundary is selected and its coordinates are transmitted. The encoder then moves along the boundary of the region and, at each step, transmits a symbol representing the direction of this movement. This continues until the encoder returns to the starting position, at which point the blob has been completely described, and encoding continues with the next blob in the image. For handwriting recognition, an 8-direction Freeman Chain Code is used to represent the time-series data of the stroke. The Freeman code carries connectivity and geometric information. Skeletal representation of features in the raster model can be expressed by the Freeman code. This code follows the contour in counter clockwise manner and keeps track of the directions as we go from one contour pixel to the next. This chaincode is used as a feature vector for the final feature vector.



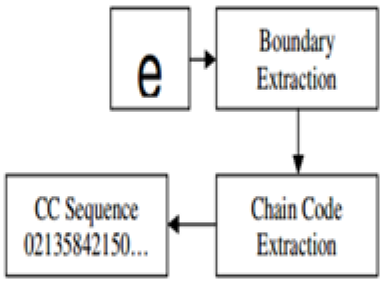


Fig: Chaincode

### 5.6.4 Principle Component Analysis (PCA)

After chaincode we take the grayscale version of the image and use standard PCA algorithm with it. This gives us a feature for every image. Over the past few years, several pattern recognition systems have been proposed based on PCA. The scheme is based on an information theory approach that decomposes training images into a small set of characteristic feature images called “eigenfaces”, which may be thought of as the principal components of the training set. Concept of PCA was introduced first by Sirovich and Kirby. Matthew Turk and Alex Pentland expanded the idea to face recognition. Training images are encoded by a small set of weights corresponding to their projection onto the new coordinate system, and are recognized by comparing them with those of known individuals.

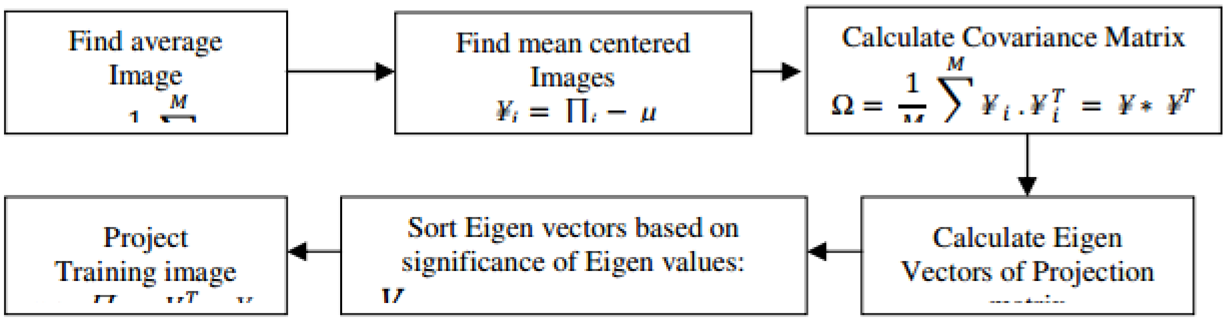

Find average Image
Find mean centered Images
$¥_i = \prod_i - \mu$
Calculate Covariance Matrix
$\Omega = \frac{1}{M}\sum^{M} ¥_i . ¥_i^T = ¥ * ¥^T$
Calculate Eigen Vectors of Projection
Sort Eigen vectors based on significance of Eigen values:
Project Training image

Fig: Feature Extraction using PCA

Here,

- µ is average of training images
- $¥_i$ is *i*th mean centered images
- Ω is covariance matrix of training dataset
- U is the set of eigenvectors associated with the Eigen values λ.

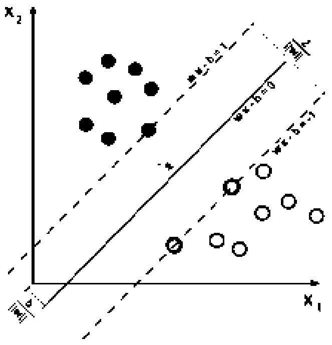
Variance and standard deviation measures the spread of the data in given data set. Nevertheless, both of this measure operates on single dimension. Covariance finds the relation between dimensions for multidimensional data. Eigen vectors with some significant Eigen values are used in pattern approximation.

### 5.6.5 Final Feature vector

Now that we have got our three different feature vector, we combine them in a single feature. This feature is then fed to the multi class SVM classifier.

# 6 Classification

As for our classifier we have used several classifier like ANN and K-Nearest Neighbor but from our results we have seen that SVM gives us the best result.



Maximum-margin hyperplane and margins for an SVM trained with samples from two classes. Samples on the margin are called the support vectors.

We know that SVM is generally for two class problem. The matlab function does not support multi class SVM. So, to solve our problem we have used an adaptation of the multisvm function by Anand Mishra which is also known as Anand SVM.

## 7 Result

We have used 50 basic characters and some compound characters for our evaluation. We have included 6 different handwriting sample for each of the characters as training set. Our training set includes handwriting variation of 6 types for each sample. We have used 10 different test samples of various people to test our system. We have also implemented some other work to compare our work with. We have got the highest accuracy for the character with basic and simple shapes. There were some misclassifications for complex looking characters ঞ, জ, ছ, ঊ. There were also misclassification for similar looking characters as well like ণ and ন, ঘ and ষ, ব and র.

| Accuracy (%) | Alphabets |
|---|---|
| <75 | ঞ, জ, ছ, ঊ |
| 75 - 80 | ঝ, ঋ, উ, ঐ, ঙ |
| 81 - 85 | ণ,ন, ঘ,ষ, র, ট, ড়, ঢ়, এ, ও, ঔ, স্ত, ক্ষ |
| 86 - 90 | য়, চ, ত, দ, ঠ, থ, ধ, শ, স, প, ফ, য, ভ, ম, হ, ঈ, ই, ন্দ, ব্দ |
| >90 | ক, খ, গ, ব, ল, ড, ঢ, ৎ, ◌ং, ◌ঃ, ◌ঁ, অ, ◌া, |

Table: Test Result

We have also implemented a few existing methods. The comparison with them is given below.

| **Methods** | Syntactic | Chaincode | PCA | Proposed |
|---|---|---|---|---|
| **Accuracy (%)** | 84 | 76 | 82 | **86.91** |

Table: Comparison with existing methods

## 8 Future work

Though our research work has showed improved performance, there can be done a lot improvement. There are some complexities that can be handled for future research on this topic. Such a complexity can be working with complex structured compound characters. For such compound characters we didn't get expected accuracy. So this can be an extension for our topic in future research.

## 9 Conclusion

Optical character recognition is such a research topic which is still developing. Due to variances in different style of writing, handwritten character recognition is a complex system to build. In our research so far, we have encountered some interesting methodologies related to different aspects of handwritten character recognition. We have implemented a method for offline character recognition with feature vectors that is combined of stroke detection, chaincode sequence and principal component of the character and classified the character using multiclass SVM approach. We got better result than the mentioned methods that were suggested in the corresponding research works.

# 10 References

- Sekhar Mandal, Sanjib Sur, Avishek Dan, Partha Bhowmick, “Handwritten Bangla Character Recognition In Machine-Printed Forms Using Gradient Information And Haar Wavelet”, 2011 International Conference On Image Information Processing (ICIIP 2011)
- Nibaran Das, Brindaban Das, Ram Sarkar, Subhadip Basu, Mahantapas Kundu, Mita Nasipuri, “Handwritten Bangla Basic And Compound Character Recognition Using MLP And SVM Classifier”, Journal Of Computing, Volume 2, Issue 2, February 2010, ISSN 2151-9617
- Suruchi G. Dedgaonkar, Anjali A. Chandavale, Ashok M. Sapkal, “Survey Of Methods For Character Recognition”, International Journal Of Engineering And Innovative Technology (IJEIT), Volume 1, Issue 5, May 2012, ISSN: 2277-3754
- Atallah Mahmoud,Al-Shatnawi And Khairuddin Omar, “Skew Detection And Correction Technique For Arabic Document Images Based On Centre Of Gravity”, Journal Of Computer Science 5 (5): 363-368, 2009, ISSN 1549-3636
- U. Pal And Sagarika Datta, “Segmentation Of Bangla Unconstrained Handwritten Text”, Proceedings Of The Seventh International Conference On Document Analysis And Recognition (ICDAR 2003)
- Zaidi Razak, Khansa Zulkiflee, Mohd Yamani Idna Idris, Emran Mohd Tamil, Mohd Noorzaily Mohamed Noor, Rosli Salleh, Mohd Yaakob @ Zulkifli Mohd Yusof, And Mashkuri Yaacob, “Offline Handwriting Text LineSegmentation : A Review”, IJCSNS International Journal Of Computer Science And Network Security, Vol.8 No.7, July 2008
- Md. Abdur RahmanAnd Abdulmotaleb El Saddik, “Modified Syntactic Method To Recognize Bengali Handwritten Characters”, IEEE Transactions On Instrumentation And Measurement, Vol. 56, No. 6, December 2007
- Vikas J Dongre, Vijay H Mankar, “Devnagari Document Segmentation Using Histogram Approach”, International Journal Of Computer Science, Engineering And Information Technology (IJCSEIT), Vol.1, No.3, August 2011
- Ram Sarkar, Nibaran Das, Subhadip Basu, Mahantapas Kundu, Mita Nasipuri, Dipak Kumar Basu, “Cmaterdb1: A Database Of Unconstrained Handwritten BanglaAnd Bangla–English Mixed Script Document Image”,IJDAR (2012) 15:71–83
- Soumen Bag, Gaurav Harit, “A Survey on Optical Character Recognition for Bangla and Devanagari Scripts”, Sadhana Vol. 38, Part 1, February 2013, Pp. 133–168
- Gunvantsinh Gohil, Rekha Teraiya, Mahesh Goyani, “Chain Code And Holistic Features Based Ocr System For Printed Devanagari Script Using Annand Svm”, International Journal Of Artificial Intelligence & Applications (IJAIA), Vol.3, No.1, January 2012
- A. Bishnu And B. B. Chaudhuri, “Segmentation Of Bangla Handwritten Text Into Characters By Recursive Contour Following”, IEEE
- B. B. Chaudhuri And U. Pal, “A Complete Printed Bangla Ocr System”,Pattern Recognition, Vol. 31, No. 5, Pp. 531Ð549, 1998