CHAPTER 1

INTRODUCTION

1.1 What is OCR?

Optical character recognition, usually abbreviated to **OCR**, is the mechanical or electronic translation of scanned images of handwritten, typewritten or printed text into machine-encoded text. [1]

It is an amazing technology that makes it possible for one to convert various types of documents ranging from PDF files to images and scanned paper documents among others into data that is searchable and editable.

Just imagine a paper document, a brochure, or a magazine article, that you need to work with. Of course, you can scan it, but this is not enough if you want to edit the information. However, there are ways to extract and repurpose all the data from the scanned document, if you use an OCR software. [2]

1.2 Necessity of Bangla OCR:

- Bangla is ranked 5th as speaking language in the world. With the digitization of every field, it is now necessary to digitized huge volume of old Bangla book by using an efficient Bangla OCR.

- Now a days we can’t extract information from old newspapers, because they are in a form of paper or in a scanned image. We can’t search by a keyword in those papers. But if we can digitized these papers we have an overall access on the information of newspapers.

- There are many valuable books (for example: we have a huge collection of books related 1971 war and Bangla language movement in SUST Muktizuddho corner) that we want to spread over Internate so that these books are easily available to the people.

- By digitized Bangla old newspapers, books we can get huge number of Bangla Data that can be used in many research related Bangla language.

- An automated Robot need to recognize any Bangla command from an image it has also need a Bangla OCR. For example: an automated Taxi Driver have to recognize the meaning of “১০০ হাত দূরে থাকুন” from the backside of a Bangladeshi truck to avoid accidents.

Chapter 2

OUR PREVIOUS WORK

Last Semester our main task was segmentation, where we had segmented each word to separate each characters from the word. In previous report we have briefly discussed some basic concepts related to Bangla OCR. These the main topics of previous report:

- Background Study
  - Chapter 2 of previous report
- Properties of Bangla font
  - Chapter 3 of previous report
- Methodology
- Segmentation

Now here we have summarized some of our previous work.

2.1 Basic Steps of Bangla OCR

The development procedure of an OCR is different for different languages. Because every languages font have their own properties. Now the most basic steps of Bangla OCR has the following particular processing steps [10] which also described in figure 4(a).

- Scanning.
- Preprocessing.
- Feature extraction or pattern recognition.
- Recognition using one or more classifier.
- Contextual verification or post processing

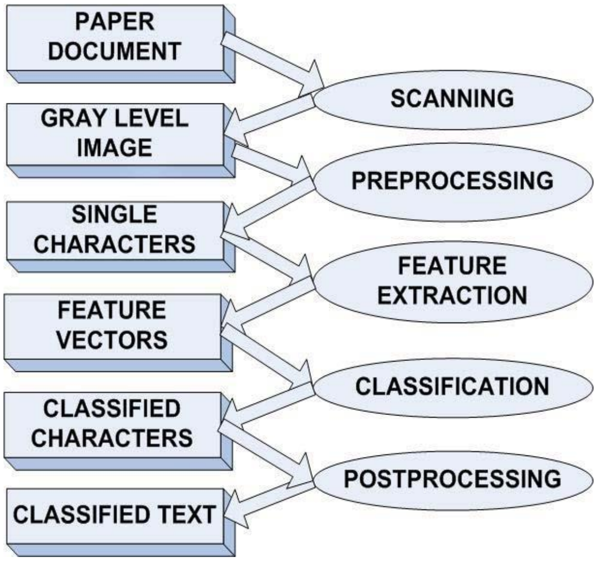


Figure 2.1(a): Steps of an OCR

2.2 Segmentation

The structure of Bangla Character is more complicated, as a result, Segmentation step is more challenging. We have used the following procedure to do segmentation with the existing approach along with our new procedure.

- ❑ Line Segmentation
- ❑ Word Segmentation
- ❑ Character Segmentation
    - Matra Line Detection [Line-Matra and Word-Matra range separately]
    - Detection of working-zone of the word (beneath the Matra-Line) into three Sub-Region
    - Upper- working zone
    - Middle- working Zone
    - Lower working Zone
    - Straight-line segmentation

- Curve-line segmentation

The most important and difficult part of segmentation was character segmentation.

2.2.1 Straight Line Segmentation

We previously determine the working zone of a word. It is the area from which we can take decision where segmentation should be take place. Since matraline connects the characters together to form a word, it is ignored during the character segmentation process to get them topologically disconnected [15]. A word constructed with basic characters is segmented into characters in a way by scanning vertically, starting from just beneath the lower row of the matraline to the baseline, considering a column of continuous white pixels as the separator, shown in Figure 5.3(e), between the characters [14].

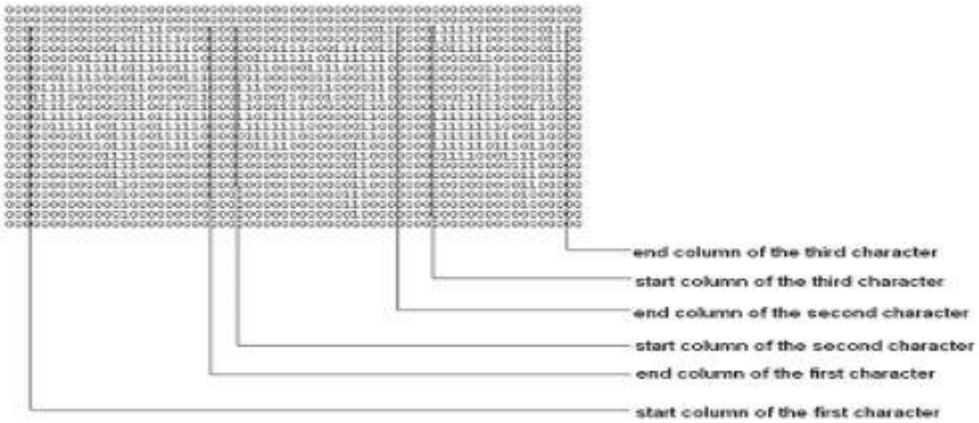

end column of the third character
start column of the third character
end column of the second character
start column of the second character
end column of the first character
start column of the first character

Figure 2.2.1(a): Straight Line segmentation

But there are some problem in straight line segmentation. For the character গ and ণ straight line segmentation split them into two small pieces. We have completely solved the problem of ণ and partially for ‘গ’. To solve this problem we slightly change the method of straight line segmentation. As In our approach segmentation takes place only on matra line and we detect matra line globally for a line as well as locally for each word. So matra line is detected very precisely. The problem of ণ has been solved.

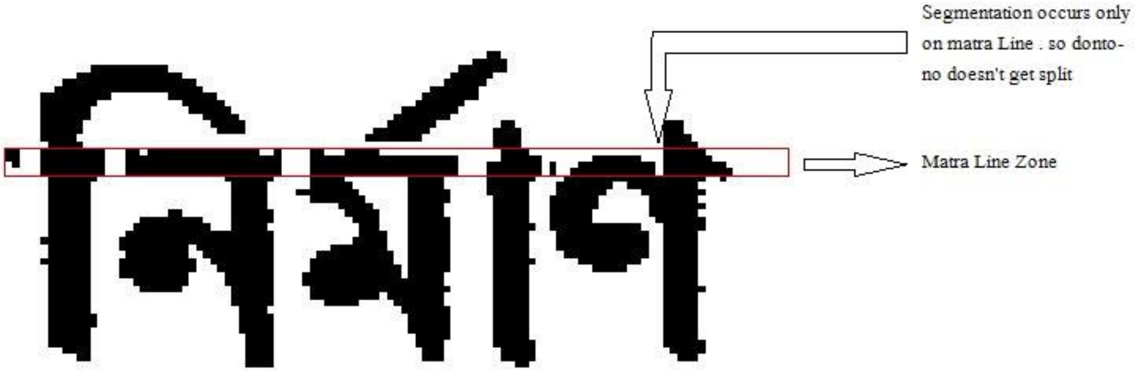

Segmentation occurs only on matra Line . so donto-no doesn't get split
Matra Line Zone

Figure 2.2.1(b): Problem Of ‘ণ’ has been solved

But In the case of character ‘গ’ things are more complicated. If we do straight-line segmentation

Then it gets split into two pieces. So we not only check the current column but also check the previous column. If previous column also doesn’t contain any pixel then we do segmentation on matra zone. By this method we can get character ‘Go’ in one piece. But in some cases it fails. For failure case, we will solve the problem in post process step by joining left piece and right piece to recognize as an individual character. Figure 5.3(g) shows the case of গ,ণ.

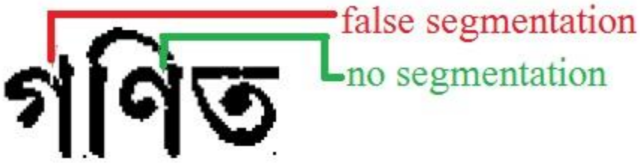


Figure 2.2.1(c): Problem of splitting গ , ণ are solved with our New Approach

There are some cases for scanned text image where straight line segmentation fails.

As examples ে+ ক, ব, র), (ছ, হ) . Figure 5.3(h) illustrates one of the failure cases of straight-line segmentation. So Special technique is needed to solve this problem. We have done some research and implemented additional technique to overcome these failure cases. Besides Straight-Line segmentation, Curve-line segmentation is that additional method which is described in next portion of this paper.

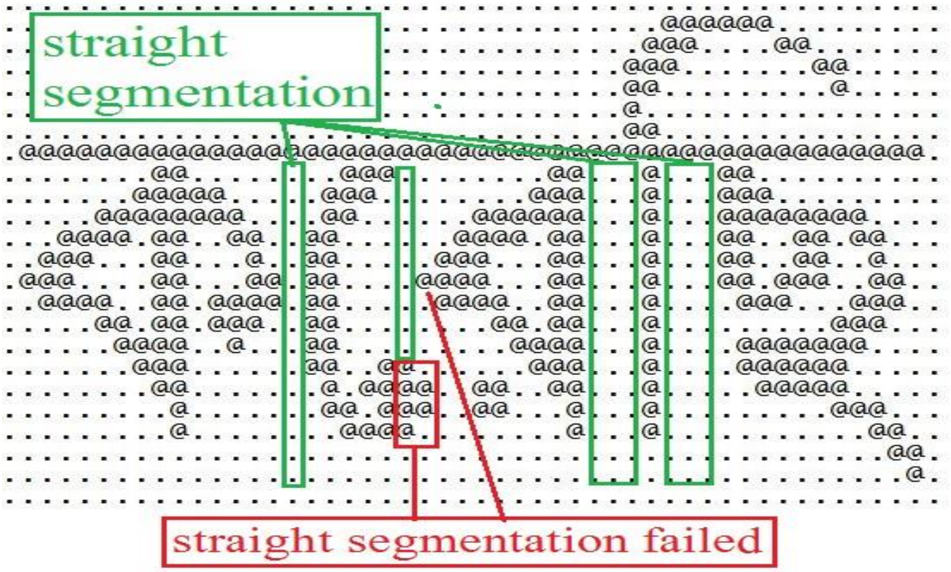


Figure 2.2.1(d): Failure case of straight line segmentation

2.2.2 Curve Line Segmentation:

There are some combinations of characters and modifiers where characters and modifiers coincide with each other. Segmentation is necessary there. But there is no single column between those characters where number of pixels is zero. Long tail of ছ, হ is another problem. So Straight-Line segmentation is not enough for this job. Curve Line segmentation is necessary here. Figure 5.3(i) shows instance of such a case.

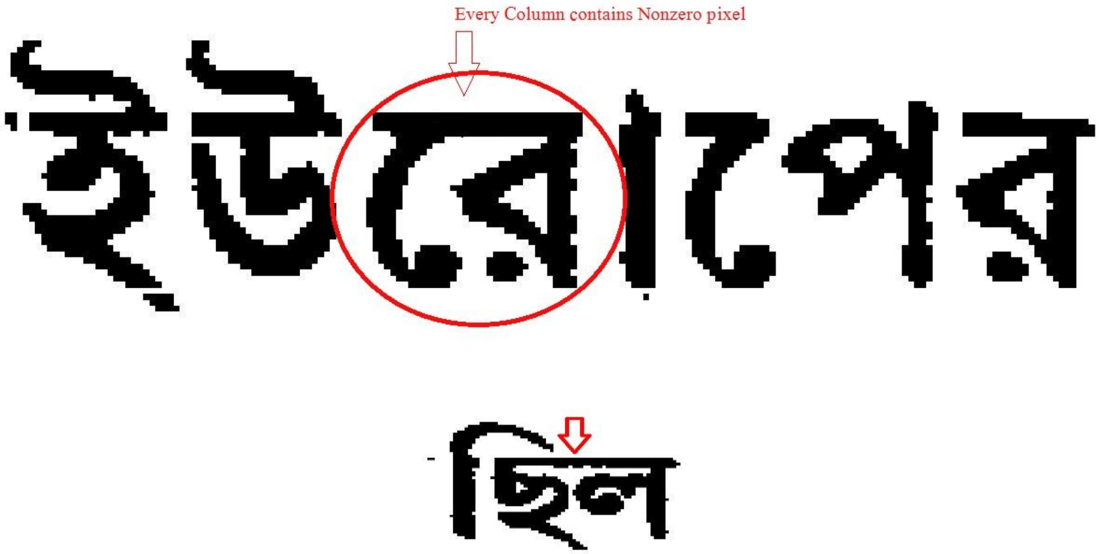
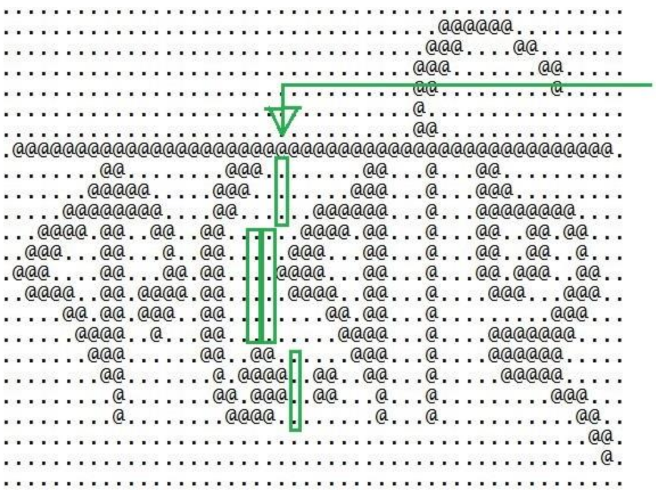


Figure 2.2.2(a): Coinciding problem of characters and modifiers

We divide the working-Zone into three sub region to implement the curve line segmentation. Those are Upper working zone (30%), Middle working zone (40%), and Lower Working zone(30%). For segmentation of nth column on matra line, instead of considering only pixel count on nth line in working zone, we consider pixel count on (n-1)th , (n-2)th, (n+1)th column in different sub working zone. First we check number of pixels on nth column upper working sub zone, then we check pixels count on (n-1)th as well as (n-2)th column middle working sub zone. Then we consider the number of pixels on (n+1)th column on Lower working sub zone. If all those sub zone contain pixels less than specific threshold value then we do segment on matra line at nth column.



```
..................................................
...................................@@@@@@@........
..................................@@@....@@.......
.................................@@@.......@@.....
.................................@@.........@.....
.................................@................
.................................@@...............
.@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@.
........@@........@@@........@@...@...@@..........
.......@@@@@.....@@@........@@@...@...@@@.........
.....@@@@@@@@....@@......@@@@@@...@...@@@@@@@@....
...@@@@.@@..@@..@@......@@@@.@@...@...@@..@@.@@...
..@@@...@@...@..@@.....@@@...@@...@...@@..@@..@...
.@@@....@@...@@.@@....@@@@...@@...@...@@.@@@..@@..
..@@@@..@@.@@@@.@@.....@@@@..@@...@....@@@...@@@..
.....@@.@@.@@@..@@........@@.@@...@.........@@@...
......@@@@..@...@@.........@@@@...@....@@@@@@@....
.......@@@......@@..@@......@@@...@....@@@@@@.....
........@@.......@.@@@@..@@..@@...@.....@@@@@.....
.........@.......@@.@@@..@@...@...@.........@@@...
.........@........@@@@........@...@...........@@..
...............................................@@.
................................................@.
..................................................
```

Considaration of pixel number of multiple column (previous and next) on different sub-zone for segmentation

Figure 2.2.2(b): Curve Segmentation

This new approach solves the problem of straight line segmentation of বে , রে, কে , হ্, হ,

Chapter 3

SEGMENTED CHARACTER EXTRACTION

After segmentation, we get a text image file where every single character is disconnected from each other. We have designed our segmentation to make every character topologically disconnected from each other. So, it is needed to extract every individual character and modifiers for next steps such as feature extraction, recognition. There are several ways to do so such as exact those in a form of box shape or with finding connected component. We have done some research and our hypothesis was-connected component should give better output. We implemented BFS with some modification to get every individual disconnected component.

3.1 Finding Connected Components:

We have implemented BFS algorithm to find out the connected components which is actually a segmented character or modifier. For every word which was segmented in previous step we scanned vertically or column wisely. At first we scanned 1st column then 2nd, 3rd from top to bottom. If we find a black pixel which was not explored before (we keep track to each pixel that is either explored before or not). If it was explored before then we skip that pixel and continue searching. If it was not explored then we start a BFS from that pixel. Before start a BFS we check either it is on matra zone or not. If it is not on the matra zone area then there is no problem. We can start a BFS. But if the scenario is not that, it means it is on matra line. As we have done our segmentation on matra line, so some unexpected noise are created. To ignore that noises, the restriction is imposed so that no BFS can start from matra zone Figure 5.4(a) Shows that scenario.

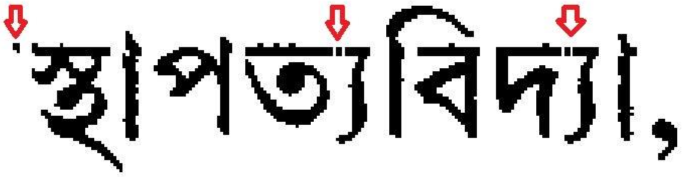


Figure 3.1(a): Unexpected noise as a result of segmentation on matra line

BFS search find out the single component. Figure 3.1(b), Figure 3.1(c) and Figure 3.1(d) Bellow illustrates that.

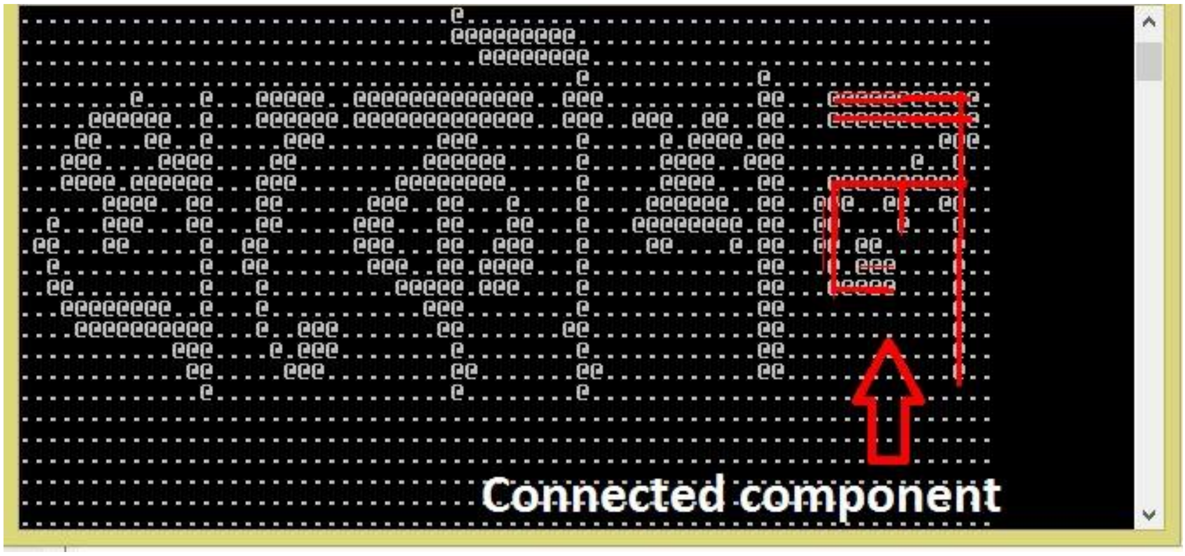
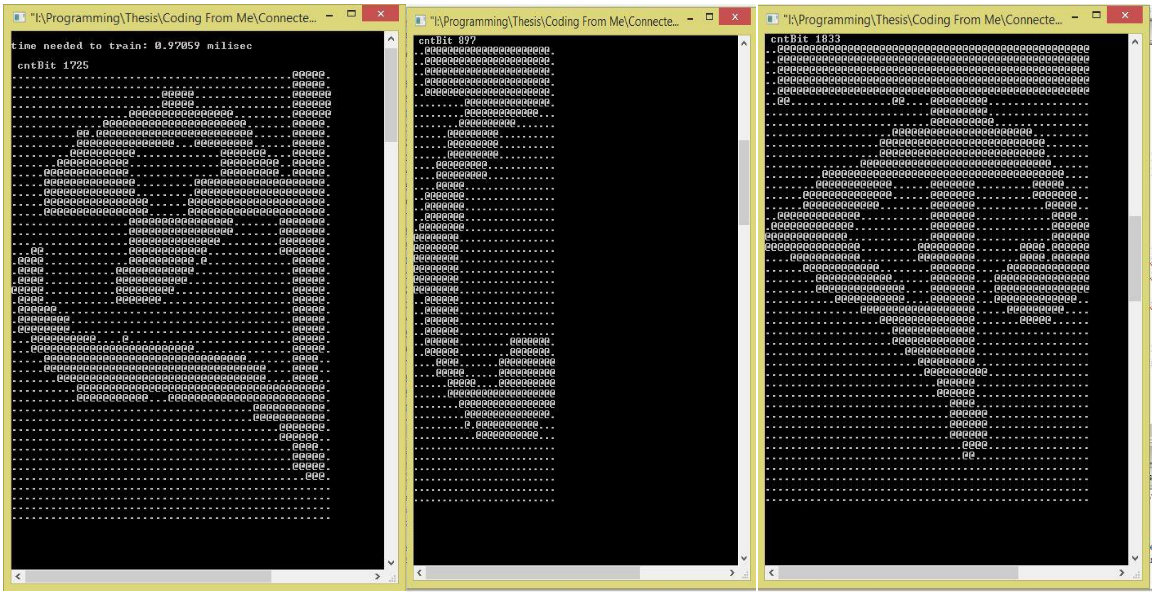


Figure 3.1(b): Extracting Segmented Character with connected component


"I:\Programming\Thesis\Coding From Me\Connecte...
"I:\Programming\Thesis\Coding From Me\Connecte...
"I:\Programming\Thesis\Coding From Me\Connecte...

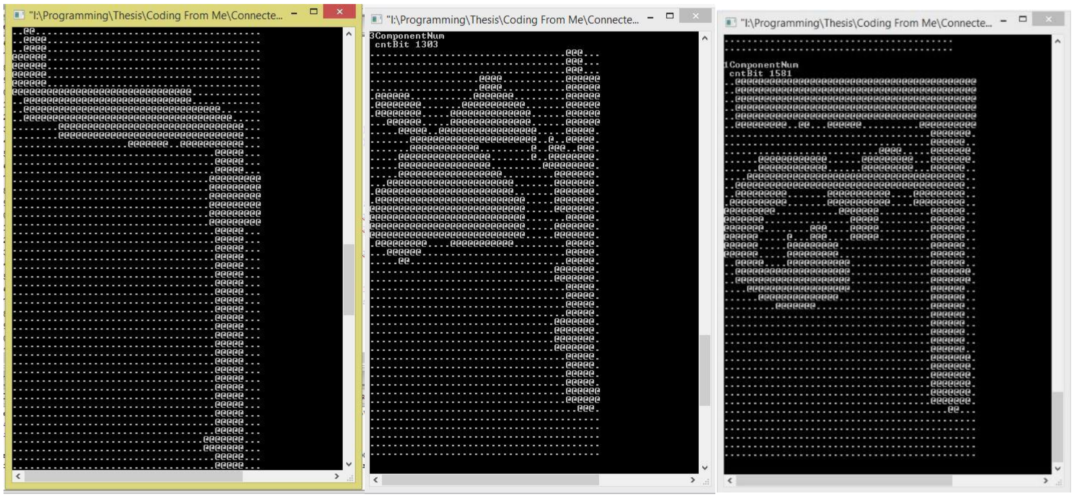
```
time needed to train: 0.97059 milisec

 cntBit 1725
```

```
cntBit 897
```

```
cntBit 1833
```

Figure 3.1(c): After Extraction of each component

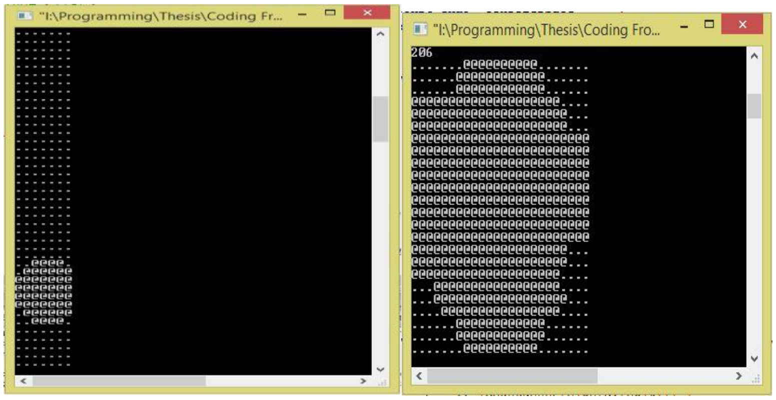

"I:\Programming\Thesis\Coding From Me\Connecte...
"I:\Programming\Thesis\Coding From Me\Connecte...
"I:\Programming\Thesis\Coding From Me\Connecte...

```
3ComponentNum
 cntBit 1303
```

```
1ComponentNum
 cntBit 1581
```

Figure 3.1(d): After Extraction of each component

For a single component, we keep track of the top most, left most, right most and bottom most point. To keep the aspect ratio and relative position of the character or modifier with the matra-line unchanged we consider that topmost point will be either ending of matra line or top of the single component. That means: Y-cordinate of topmost point = maximum (Y-cordinate of explored pixel, Y-coordinate of matra ending position) . Figure 5.4(b) shows the difference of two image where 1st image keeps same aspect relation and another doesn't:


"I:\Programming\Thesis\Coding Fr...
"I:\Programming\Thesis\Coding Fro...

```
206
```

Figure 3.1(e): Difference of Two different method of finding connected component [first one keep aspect ratio same and second one donesn't]

The second component/pattern (ফোঁটা) is hard to recognized as its structure is similar to । - দাঁড়ি।
There are several cases where keeping aspect ratio and relative position same is very important for the next steps (recognition). So we modified our finding connected component algorithm of extracting out the single component.

Chapter 4

FEATURE EXTRACTION & RECOGNITION

Optical character recognition (OCR) is one of the most successful applications of automatic pattern recognition. The performance of character recognition or pattern system is depends on proper feature extraction. The main idea of the feature extraction techniques is to identify characters based on features that are similar to the features humans use to identify characters. Humans recognize characters easily and they repeat the character recognition process thousands of times every day as they read papers and books. However, after many years of intensive and research, the main goal of developing an optical character recognition system with the same reading capabilities as humans still remains unachieved.

Developers or programmers have to manually determine the properties of characters they feel are important. Selection of a feature extraction technique is probably the single most important factor for high recognition performance in character recognition systems.

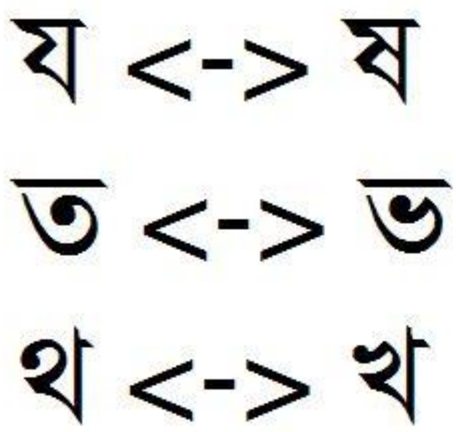
4.1 Our Observation and analysis:

We have experimented to find out image feature which is appropriate for better recognition. We have analyzed the characteristics of Bangla characters. We have tried to find out some common pattern in Bangla character. As an example left side of several Bangla characters are the same. Just a straight line. Some of these characters are shown below:

খ, ঘ, ন, য, র, ল, ধ, ণ

Figure 4.1(a): Characters with same pattern in left side area

There are several character which are almost same when we consider outer shape only. So if we only consider outer shape for recognition, then the approach fails. Some example of these types of characters are:



য <-> ষ

ত <-> ভ

থ <-> খ

Figure 4.1(b): Characters with almost similar outer shape

We have researched on these characteristics and find out some statistical features. They are

- Number of pixel in Horizontal and Vertical Histogram
- Distance from different side of image.
- Skeleton of characters.
- Number of corner in image.
- Pixel ratio in sub-zone, number of corner et

These features are described here:

4.2 Number of pixel in Horizontal and Vertical Histogram:

In this approach, the number of black pixel in each row is counted in Vertical Histogram, and number black pixel of each column is counted in Horizontal Histogram. Each row and column correspond to a value in vertical and horizontal histogram. We also count black pixel number in a diagonal and construct diagonal histogram.

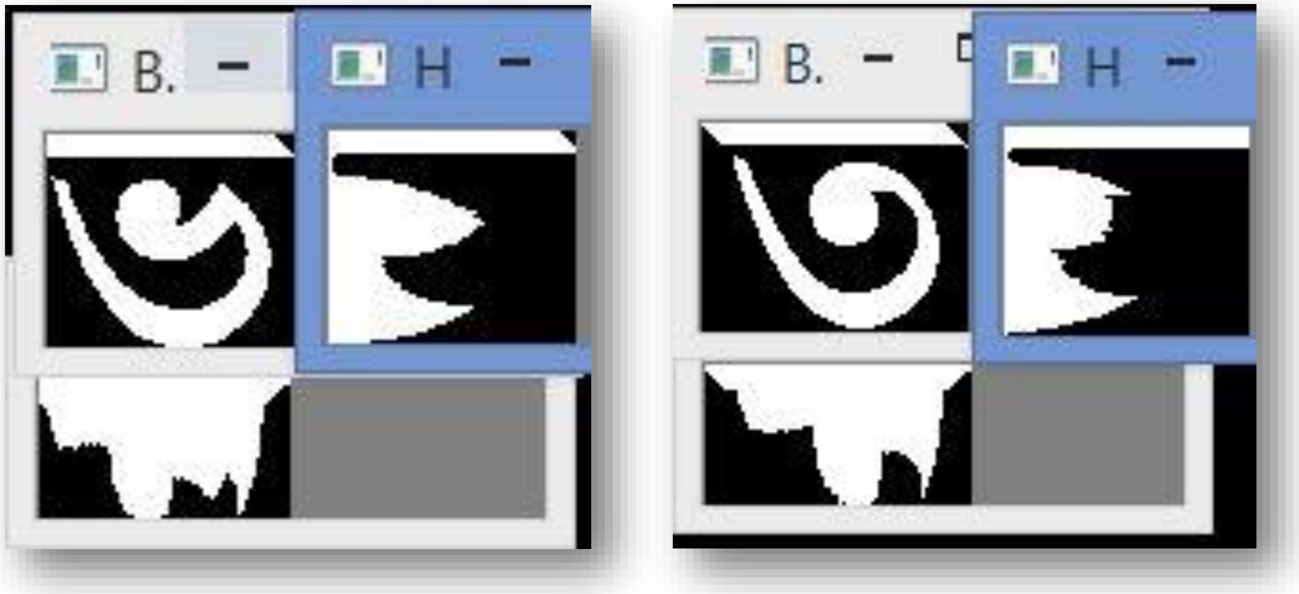


Figure 4.2(a): The case where Histogram method fails to recognize correctly

But only this approach is not enough for distinguishing between some very similar characters where black pixel number in Horizontal and Vertical histogram are very close. One of the cases is given below:

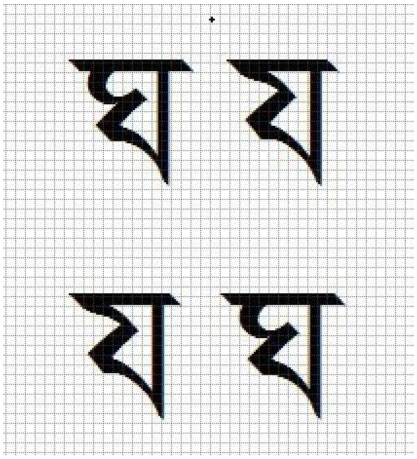


Figure 4.2(b): Similar character where each row and column contains approximately the same number of black pixels.

There are many cases where cumulative difference of horizontal and vertical histogram value between different characters are almost same. It can not differentiate between mirror image. ‘১’ and ‘ে’ is nearly mirrored image of each other . This approach fails to recognize them correctly.

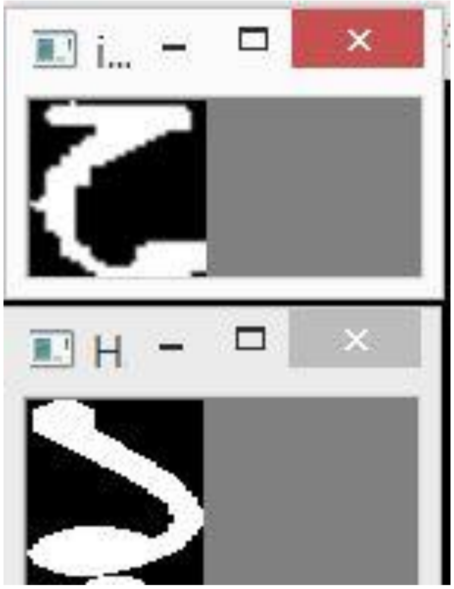


Figure 4.2(c): Histogram method fails to recognize mirrored image exactly

That is why we could not apply this method for image matching or recognition.

4.3 Distance from different side of image:

In this method it compares data through a projection. It calculate distance from left side border to first black pixel. And also from right side border. We calculate these for every row of the image. It does the same for calculating distance from top and bottom border also.

But the main problem for this method is: this method ignores an important part of the character image. It ignores the middle part of the character. Which is confined by the outer border of the character. Image at bellow is given for illustration:

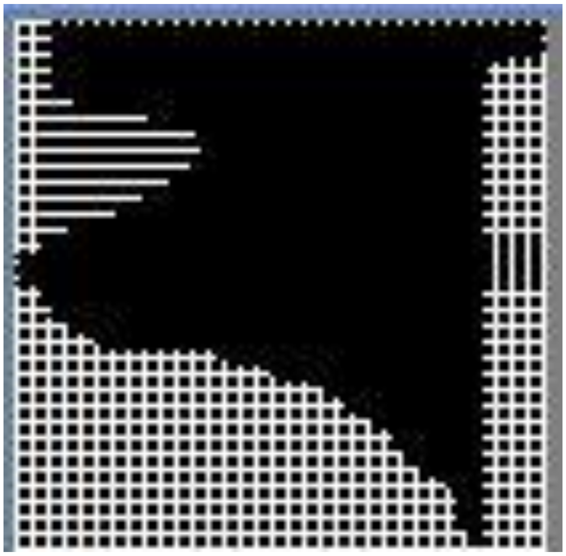


Figure 4.3(a): Distance from Border Method unable to access the middle part of any character, so middle part remain unreachable.

So this method fails to distinguish between ‘**য**’ and ‘**ষ**’ and many other cases.

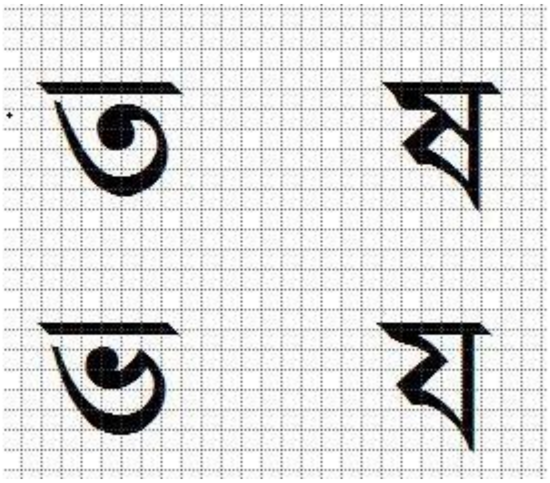


Figure 4.3(b): Some of the cases where Distance from Border approach can not recognize characters with same outer shape.

4.4 Skeleton of Characters:

In this approach, we first make the test character thinner and thicker individually. Then we compare it with train character image set. For comparing we check out that if the skeleton fits in the black pixel region of test character image. If the skeleton fits in that black pixel region 100% precisely. And if all white pixel in thicker image, remains in the white pixel region in train character image. Then the method takes decision that the images are same or matched.

To get the skeleton of a character we have used jahn suang algorithm [19].

But there are some problems in this method. When we make a character image thinner. The actual shape is distorted. And most importantly there is a “Don’t Care” zone. Where the method do not consider if the pixel is matched or not. The shape of the skeleton of **‘স’** and **‘ম’ ,** ‘য’ and **‘ফ’**, and many other cases become the same. So this idea was our wrong hypothesis.

A case where the approach fail to recognize correct character is given below. Test input is ‘য’**,** after making it thinner and thicker we got image like these:

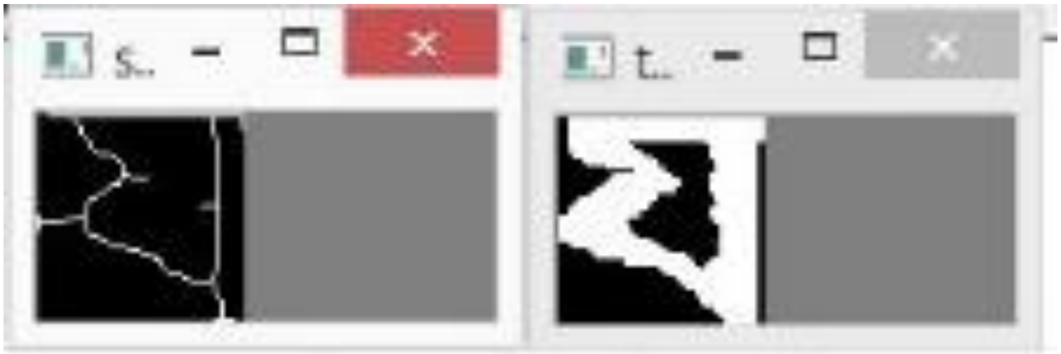


Figure 4.4(a): After making test character 'য' thinner and thicker

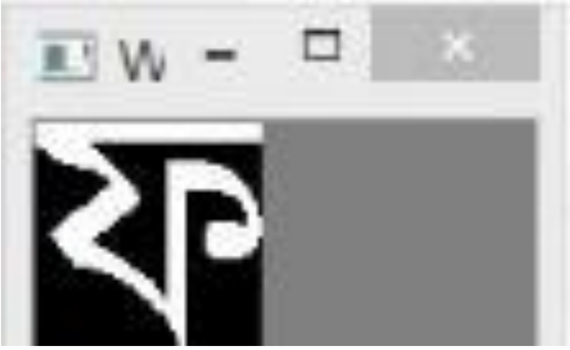


Figure 4.4(b): Recognized character from train set for 'য' with skeleton matching method [As the image were shown in console all black pixel are represented by white pixel and vice versa]

Here all the black pixel in skeleton remains in the Black pixel (white pixel in console output) zone in train image and all white pixel (black pixel in console output) in the thicker image remains in white pixel (black pixel in console) zone in train image 'ফ'. The main drawback of this approach is it's don't care zone.

4.5 Corner of Character Image:

In this approach we find out the corner of an image. Then we count the number of total corner in the image. We applied Harris-Stephens method to find out the corner of an image [18]. It works moderately well for sharp and clear image. But as we works for scanned image, where the corner of character is not so sharp, and the image can be distorted, the result of the method is not satisfactory.

There are some character where there is no corner at all for distortion.

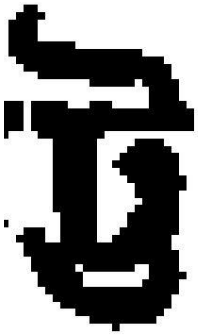
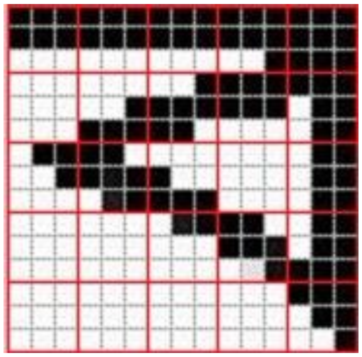


Figure 4.5(a): Character where corner detection fails.

4.6 Sub-Zoning Approach (Our final chosen approach):

In this approach, the image is divided into specific number of squared sub-zone. Size of each sub-zone is 3x3. Here 3x3 size is optimal. If we take height and width greater than or less than 3 then it is difficult to take the proper advantage of this approach.

With this approach, at first the number of black pixel in each sub-zone is calculated. The number of black pixel in each sub-zone is stored in corresponding index.



6	6	6	7	9
0	4	6	6	7
3	6	4	0	6
0	0	2	6	7
0	0	0	0	6

Figure 4.6(a): Illustration of counting black pixel in every subzone

To compare two images with this approach, we calculate the difference of number of black pixel in each sub-zone. Then we take the summation of the difference of all sub-zone's black pixel number. If we compare two image with only these value, then some ambiguous cases remain unsolved. Such as case is recognition problem in 'ব' and 'য':

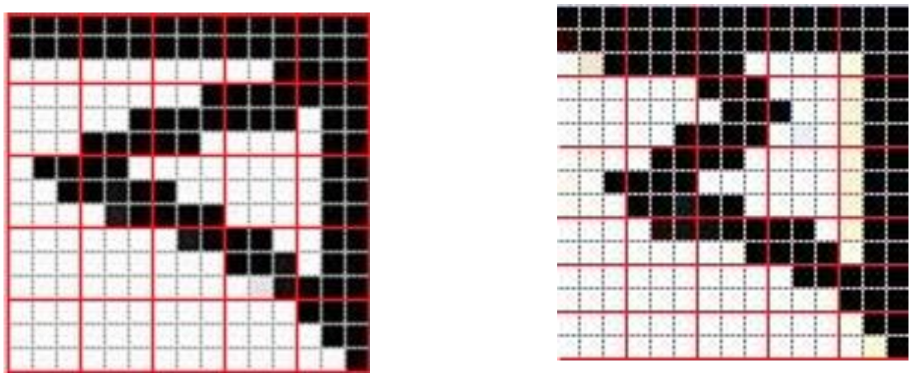


Figure 4.6(b): Ambiguous case of '**ব**' and '**য**'

So we implement another idea. We consider a sub-zone is different in two image if the difference of number of black pixel in corresponding sub-zones in those two images is greater than a threshold value. That means two sub-zone is different if 80-90% of total pixel is different in that sub-zone. So if two sub-zone is totally mismatched then we consider them different. Otherwise we consider it as matched sub-zone.

Then we count the number of subzone which are different or dissimilar. [We have determined which sub-zone are different by a threshold value]. Then we multiply that number with a weight to make a 50%-50% distribution.

So the equation for calculating final value for two images by which we compare those images will be:

$$\boldsymbol{val} = (\ \sum(\boldsymbol{difference\ of\ the\ number\ of\ black\ pixel\ in\ each\ sub\ zone}\ ) \\ + \boldsymbol{Weight} * (\boldsymbol{number\ of\ mismatched\ sub\_zone})$$

By this approach, if the extracted or test image is slightly distorted, the method can recognize correctly as we do not compare images with single pixel by pixel comparison. Rather we compare images with sub region.

After some observation, it is figured out that one of the major problem in our Bangla character recognition approach is: if the same image is shifted 2-3 bit left or right then our approach fails to recognize them correctly. And this shifting occurs frequently within segmentation and character extraction phase. That scenario is illustrated by the image bellow:

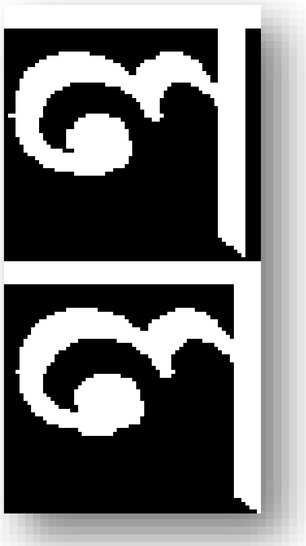


Figure 4.6(c): Shifting Problem if upper 'ল' is shifted to right 3 bits then the two image would be exactly same.

So, we modify our approach to get more precise result. We shift the test or extracted image 1 bit, 2 bit left and right. And the whole procedure repeat for the new shifted image. We choose the best candidate from this shifted image which is the closest to the train characters.

By following this procedure, we can get more precise result.

Chapter 5

Post Processing

Recognizing a character correctly is not the last work to get output. We have to merge this recognized characters in a sequence to get the actual word. Post processing is the technique for reducing the error after recognition part. Post processing of Optical Character Recognition requires many steps. This steps are:

- Writing the recognized characters in the right sequence
- Spell check for word correction
- Sentence correction

5.1 Correction of the writing sequence:

This case is special for Bangla characters. For English language after recognizing the characters if we write the characters in the same sequence we get them at the time of extracting there will be no change in output. Suppose there's an image containing English word 'Bil'. After extracting each character individually they are passed to recognition module then after recognition they are put at the same sequence to make this word.

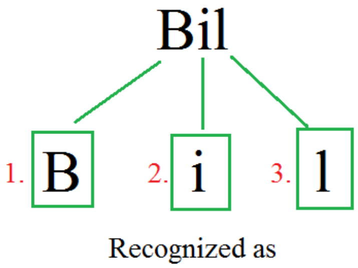


Fig 5.1(a): Sequence of Recognition

But in the case of Bangla character if the characters are written in the same sequence the scenario is not the same as English. This case is illustrated by the example: an image containing word ‘বিল’.

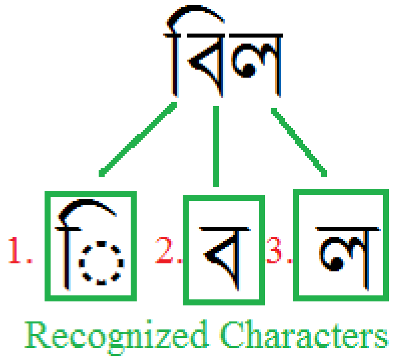


Fig 5.1(b): Sequence of Recognized Characters

Now after recognition if we write these characters in the same sequence then we get a text like that – ‘িবল’. Because ‘ি’ should write after ‘ব’ to get ‘বি’.

The cases for which we have to switch the writing sequence:

1. While scanning we get ‘ে’, ‘ি’, ‘ৈ’ modifier before the character. But in writing text file we need to put modifier after the character.
2. We need multiple characters and modifiers swapping when the case is like: ‘কর্ডোভা’

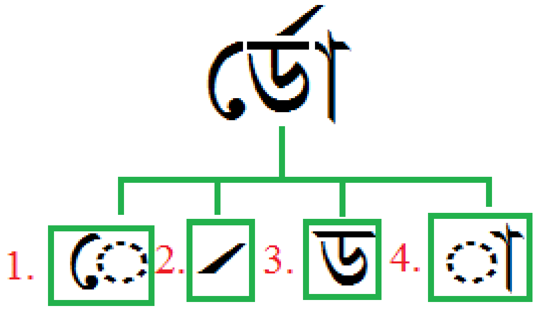


Figure 5.1(c): Sequence of Recognized Characters

Here to get 'র্ডো' first to write 3rd character 'ড' then 2nd character 'রেফ' then 1st character 'ে' at last the 4th character 'া'. So the things get more and more complecated .

3. As we extract the characters using connected component sometimes we failed to get the whole character as some characters have many component which are disconnected. Such as: 'র' 'য়' 'ঢ়' 'ড়' 'ঃ' 'ং'. We get each component separately. For example : we get 'ব' and a fota which represent 'র'.

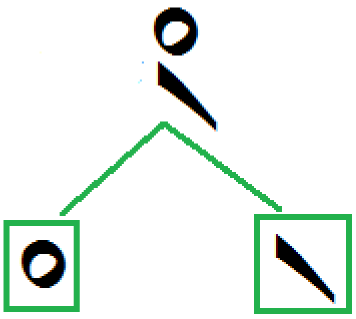
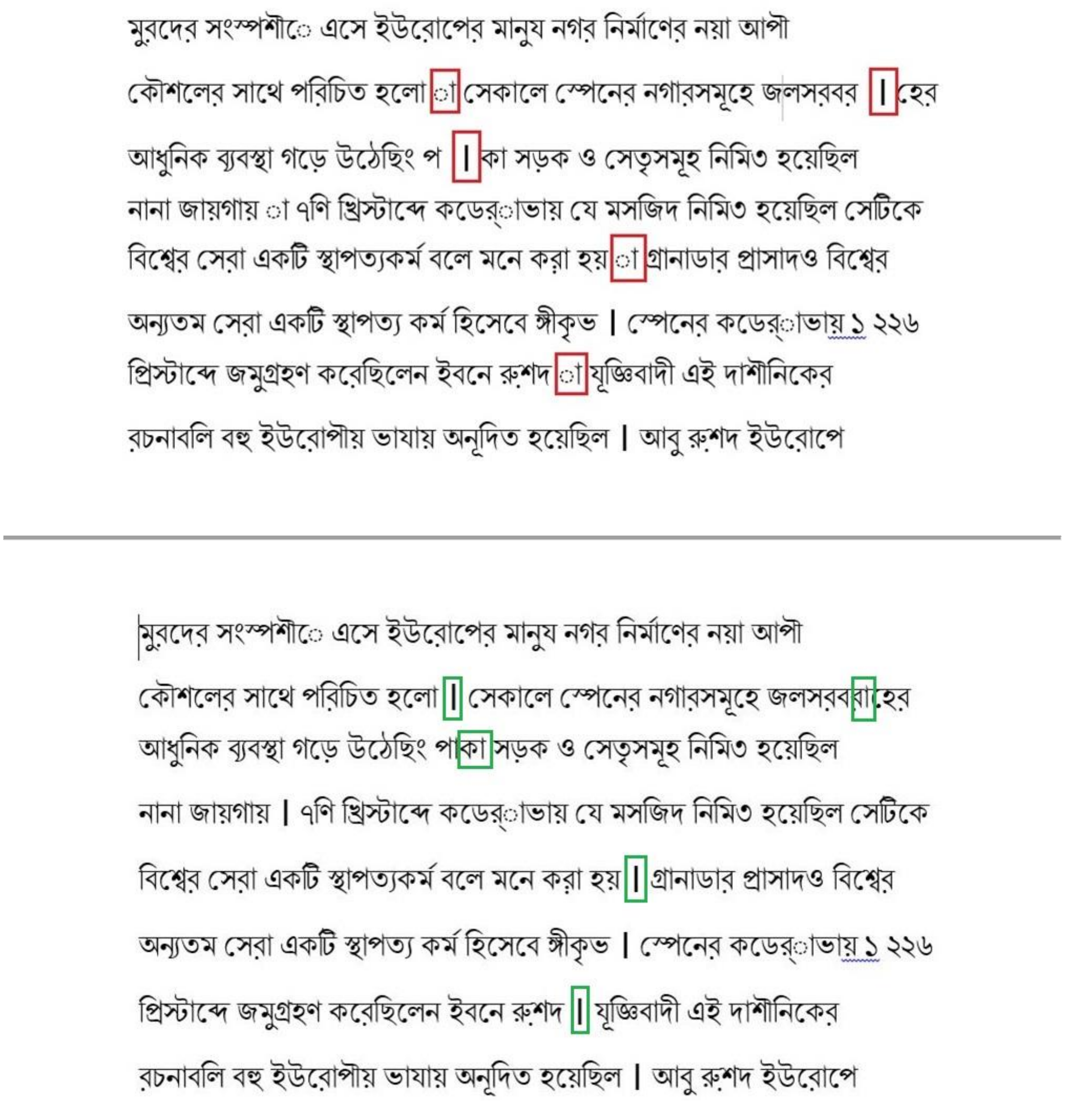


Figure 5.1(d): Disconnected components

For this character 'ং' we get two separate figure. So when we recognize this two figure subsequently we can say that this is the character 'ং' .

4. দাড়ি– '।' and আ–কার 'া' are so identical that it is very hard to distinguish them. So we check if any 'দাড়ি' is recognized in the middle of word, it must be 'আ–কার'. Or if 'আ–কার' is the only character of a single word. Then it must be a 'দাড়ি'.



মুরদের সংস্পর্শে এসে ইউরোপের মানুয নগর নির্মাণের নয়া আপী
কৌশলের সাথে পরিচিত হলো ◌া সেকালে স্পেনের নগারসমূহে জলসরবর । হের
আধুনিক ব্যবস্থা গড়ে উঠেছিং প । কা সড়ক ও সেতৃসমূহ নিমিত হয়েছিল
নানা জায়গায় ◌া ৭ণি খ্রিস্টাব্দে কডের্◌াভায় যে মসজিদ নিমিত হয়েছিল সেটিকে
বিশ্বের সেরা একটি স্থাপত্যকর্ম বলে মনে করা হয় ◌া গ্রানাডার প্রাসাদও বিশ্বের
অন্যতম সেরা একটি স্থাপত্য কর্ম হিসেবে স্বীকৃভ । স্পেনের কডের্◌াভায় ১ ২২৬
প্রিস্টাব্দে জমুগ্রহণ করেছিলেন ইবনে রুশদ ◌া যূক্তিবাদী এই দার্শনিকের
রচনাবলি বহু ইউরোপীয় ভাযায় অনূদিত হয়েছিল । আবু রুশদ ইউরোপে

মুরদের সংস্পর্শে এসে ইউরোপের মানুয নগর নির্মাণের নয়া আপী
কৌশলের সাথে পরিচিত হলো । সেকালে স্পেনের নগারসমূহে জলসরবরা হের
আধুনিক ব্যবস্থা গড়ে উঠেছিং পাকা সড়ক ও সেতৃসমূহ নিমিত হয়েছিল
নানা জায়গায় । ৭ণি খ্রিস্টাব্দে কডের্◌াভায় যে মসজিদ নিমিত হয়েছিল সেটিকে
বিশ্বের সেরা একটি স্থাপত্যকর্ম বলে মনে করা হয় । গ্রানাডার প্রাসাদও বিশ্বের
অন্যতম সেরা একটি স্থাপত্য কর্ম হিসেবে স্বীকৃভ । স্পেনের কডের্◌াভায় ১ ২২৬
প্রিস্টাব্দে জমুগ্রহণ করেছিলেন ইবনে রুশদ । যূক্তিবাদী এই দার্শনিকের
রচনাবলি বহু ইউরোপীয় ভাযায় অনূদিত হয়েছিল । আবু রুশদ ইউরোপে

Figure 5.1(e): Before and after processing 'আ–কার' and 'দাঁড়ি'

5.2 Word Correction & Sentence Correction:

When an OCR system fails to recognize a character, an OCR error is produced, commonly causing a spelling mistake in the output text. For example: if ষ is recognized as য then the word ‘কৃষক’ becomes ‘কৃযক’. But the second one is not a correct Bangla word. So we can easily correct it by spell checking.

But sometimes wrong recognition can give us correct word. For example: if ‘খ’ is recognized as ‘থ’ then ‘খাল’ can be recognized as ‘থাল’. In that case we may have to correct the sentence for getting actual output.

Word correction and Sentence correction is a huge field for research. We expect in future thesis team will be formed for researching in those fields.

Chapter 6

RESULT ANALYSIS AND DISCUSSIONS

Here we have discussed about our experiments performance and compare our performance with previous work.

6.1 Error Rate of Recognition part:

Our recognition technique can recognize about 95% characters correctly. For some characters that have a similar structure have failed to recognize correctly. This cases are shown below.

Table (1): Error characters

Original character	Recognized as
খ	থ
ষ	য
ত	ভ
ন্স	ন্দ
ক্ষ	ম্ম

শ্	শ্র

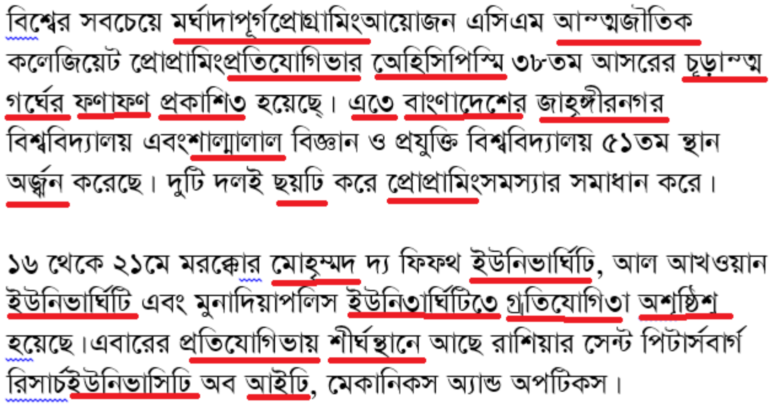
We have tested our program for some images and counted number of error words.

Table (2): Number of Error For Different Scanned Image

Image	Number of words	Number of words failed to recognize
Image 1	325	39
Image 2	232	22
Image 3	210	26
Image 4	172	17
Image 5	136	15

6.2 Compare with previous work

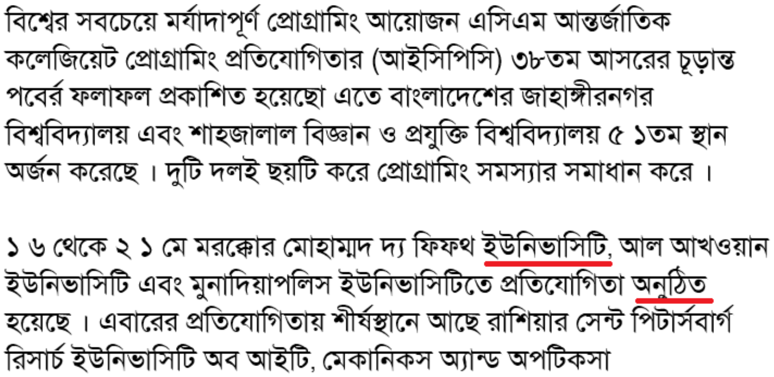
We have compared present OCR’s output with previous years OCR. At first we tested the output for an ideal image. (Image that is not scanned through scanner. We just convert a typed text into image. So there was no noise in the image)



বিশ্বের সবচেয়ে মর্ঘাদাপূর্গপ্রোগ্রামিংআয়োজন এসিএম আন্ত্মর্জাতিক কলেজিয়েট প্রোপ্রামিংপ্রতিযোগিভার অেহিসিপিস্মি ৩৮-তম আসরের চূড়ান্ত্ম গর্ঘের ফণাফণ প্রকাশিত হয়েছে। এতে বাংণাদেশের জাহৃঙ্গীরনগর বিশ্ববিদ্যালয় এবংশাল্মালাল বিজ্ঞান ও প্রযুক্তি বিশ্ববিদ্যালয় ৫১তম স্থান অর্জ্বন করেছে। দুটি দলই ছয়ঢি করে প্রোপ্রামিংসমস্যার সমাধান করে।

১৬ থেকে ২১মে মরক্কোর মোহৃম্মদ দ্য ফিফথ ইউনিভার্ঘিঢি, আল আখওয়ান ইউনিভার্ঘিটি এবং মুনাদিয়াপলিস ইউনিতার্ঘিটিতে গ্রতিযোগিতা অশৃষ্ঠিশ হয়েছে।এবারের প্রতিযোগিভায় শীর্ঘস্থানে আছে রাশিয়ার সেন্ট পিটার্সবার্গ রিসার্চইউনিভাসিঢি অব আইঢি, মেকানিকস অ্যান্ড অপটিকস।

Figure 6.2(a): Previous Output for an ideal image

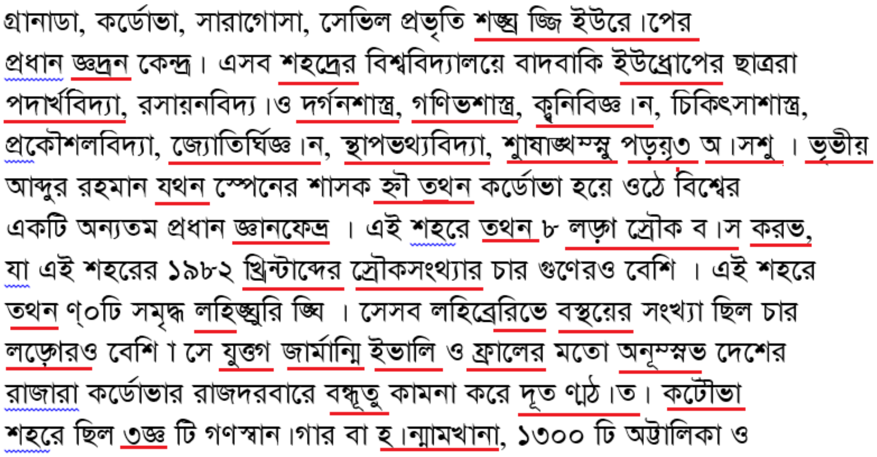


বিশ্বের সবচেয়ে মর্যাদাপূর্ণ প্রোগ্রামিং আয়োজন এসিএম আন্তর্জাতিক কলেজিয়েট প্রোগ্রামিং প্রতিযোগিতার (আইসিপিসি) ৩৮-তম আসরের চূড়ান্ত পর্বের্র ফলাফল প্রকাশিত হয়েছো এতে বাংলাদেশের জাহাঙ্গীরনগর বিশ্ববিদ্যালয় এবং শাহজালাল বিজ্ঞান ও প্রযুক্তি বিশ্ববিদ্যালয় ৫ ১তম স্থান অর্জন করেছে । দুটি দলই ছয়টি করে প্রোগ্রামিং সমস্যার সমাধান করে ।

১ ৬ থেকে ২ ১ মে মরক্কোর মোহাম্মদ দ্য ফিফথ ইউনিভাসিটি, আল আখওয়ান ইউনিভাসিটি এবং মুনাদিয়াপলিস ইউনিভাসিটিতে প্রতিযোগিতা অনুঠিত হয়েছে । এবারের প্রতিযোগিতায় শীর্ষস্থানে আছে রাশিয়ার সেন্ট পিটার্সবার্গ রিসার্চ ইউনিভাসিটি অব আইটি, মেকানিকস অ্যান্ড অপটিকসা

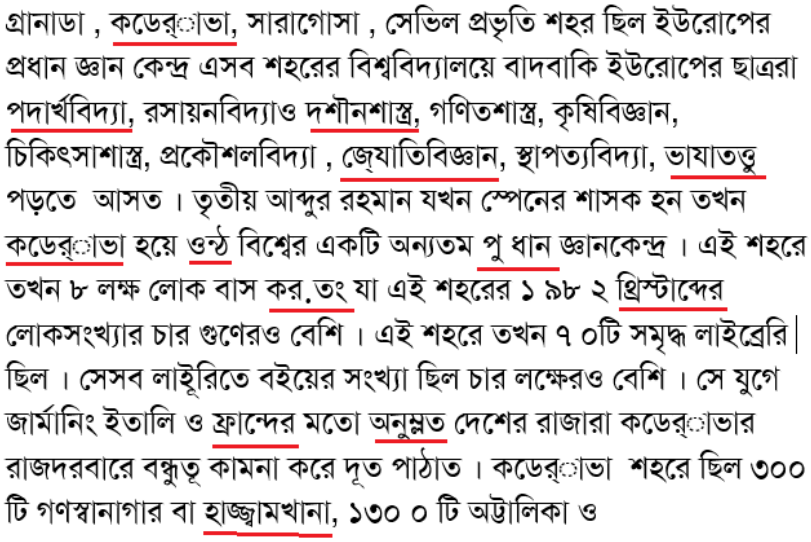
Figure 6.2(b): Present Output for an ideal image

Then we have tested for scanned image. Output is shown below.



গ্রানাডা, কর্ডোভা, সারাগোসা, সেভিল প্রভৃতি শজ্ম্র জ্জি ইউরে।পের
প্রধান জ্ভ্রন কেন্দ্র। এসব শহদ্রের বিশ্ববিদ্যালয়ে বাদবাকি ইউধ্রোপের ছাত্ররা
পদার্থবিদ্যা, রসায়নবিদ্য।ও দর্গনশাস্ত্র, গণিভশাস্ত্র, কৃ্নিবিজ্ঞ।ন, চিকিৎসাশাস্ত্র,
প্রকৌশলবিদ্যা, জ্যোতির্ঘিজ্ঞ।ন, স্থাপভথ্যবিদ্যা, শুাষাজ্খম্স্নু পড়য়্ত অ।সশু । ভৃভীয়
আব্দুর রহমান যথন স্পেনের শাসক হ্ন্ী তথন কর্ডোভা হয়ে ওঠে বিশ্বের
একটি অন্যতম প্রধান জ্ঞানফেন্দ্র । এই শহরে তথন ৮ লড়্ক্া স্রোক ব।স করভ,
যা এই শহরের ১৯৮২ খ্রিন্টাব্দের স্রোকসংথ্যার চার গুণেরও বেশি । এই শহরে
তথন ণ্০টি সমৃদ্ধ লহিজ্ঞ্রুরি জ্বি । সেসব লহিব্রেরিভে বস্থয়ের সংখ্যা ছিল চার
লড়্ক্োরও বেশি । সে যুক্তগ জার্মানিু্ ইভালি ও ফ্রালের মতো অনৃম্স্নভ দেশের
রাজারা কর্ডোভার রাজদরবারে বন্ধৃতু কামনা করে দূত ণ্া্ঠ।ত। কটৌভা
শহরে ছিল ৩জ্ঞ টি গণস্বান।গার বা হ।ন্মামখানা, ১৩০০ টি অট্টালিকা ও

Figure 6.2(c): Previous Output for a Scanned image



গ্রানাডা , কডের্◌াভা, সারাগোসা , সেভিল প্রভৃতি শহর ছিল ইউরোপের
প্রধান জ্ঞান কেন্দ্র এসব শহরের বিশ্ববিদ্যালয়ে বাদবাকি ইউরোপের ছাত্ররা
পদার্থবিদ্যা, রসায়নবিদ্যাও দর্শীনশাস্ত্র, গণিতশাস্ত্র, কৃষিবিজ্ঞান,
চিকিৎসাশাস্ত্র, প্রকৌশলবিদ্যা , জে্যাতিবিজ্ঞান, স্থাপত্যবিদ্যা, ভাযাতত্ত্ব
পড়তে আসত । তৃতীয় আব্দুর রহমান যখন স্পেনের শাসক হন তখন
কডের্◌াভা হয়ে ওন্ঠ বিশ্বের একটি অন্যতম পু ধান জ্ঞানকেন্দ্র । এই শহরে
তখন ৮ লক্ষ লোক বাস কর.তং যা এই শহরের ১ ৯৮ ২ থ্রিস্টাব্দের
লোকসংখ্যার চার গুণেরও বেশি । এই শহরে তখন ৭ ০টি সমৃদ্ধ লাইব্রেরি|
ছিল । সেসব লাইৃরিতে বইয়ের সংখ্যা ছিল চার লক্ষেরও বেশি । সে যুগে
জার্মানিং ইতালি ও ফ্রান্দের মতো অনুম্নত দেশের রাজারা কডের্◌াভার
রাজদরবারে বন্ধুতৃ কামনা করে দূত পাঠাত । কডের্◌াভা শহরে ছিল ৩০০
টি গণস্বানাগার বা হাজ্জ্বামখানা, ১৩০ ০ টি অট্টালিকা ও

Figure 6.2(d): Present Output for a Scanned image

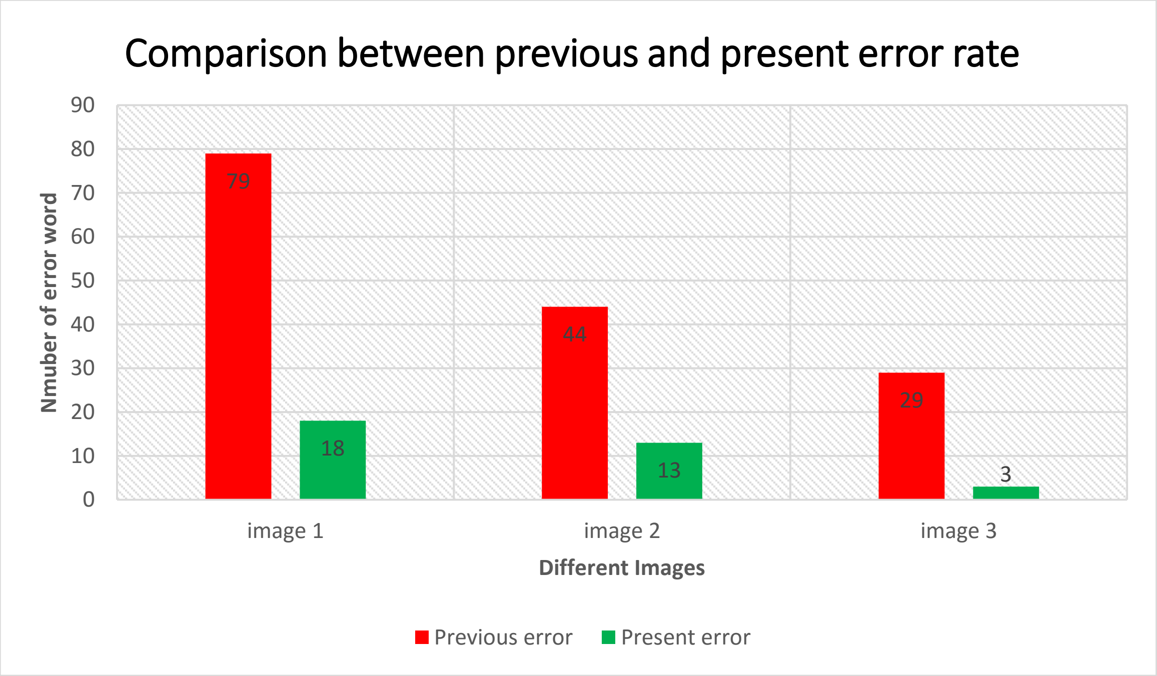


Figure 6.2(e): Comparison between previous and present error rate

6.3 Time Analysis

We have analyzed the time taken for the segmentation and recognition process for different images that contain different amount of words. Analyzed data are shown below: (all images are scanned at 300 dpi)

Table (3): Time to segmentation & recognition for Different Number Of Words

Image no	Number of words	Time count(ms)
Image 1	325	8556
Image 2	232	6224
Image 3	210	5510
Image 4	172	4364
Image 5	136	3272

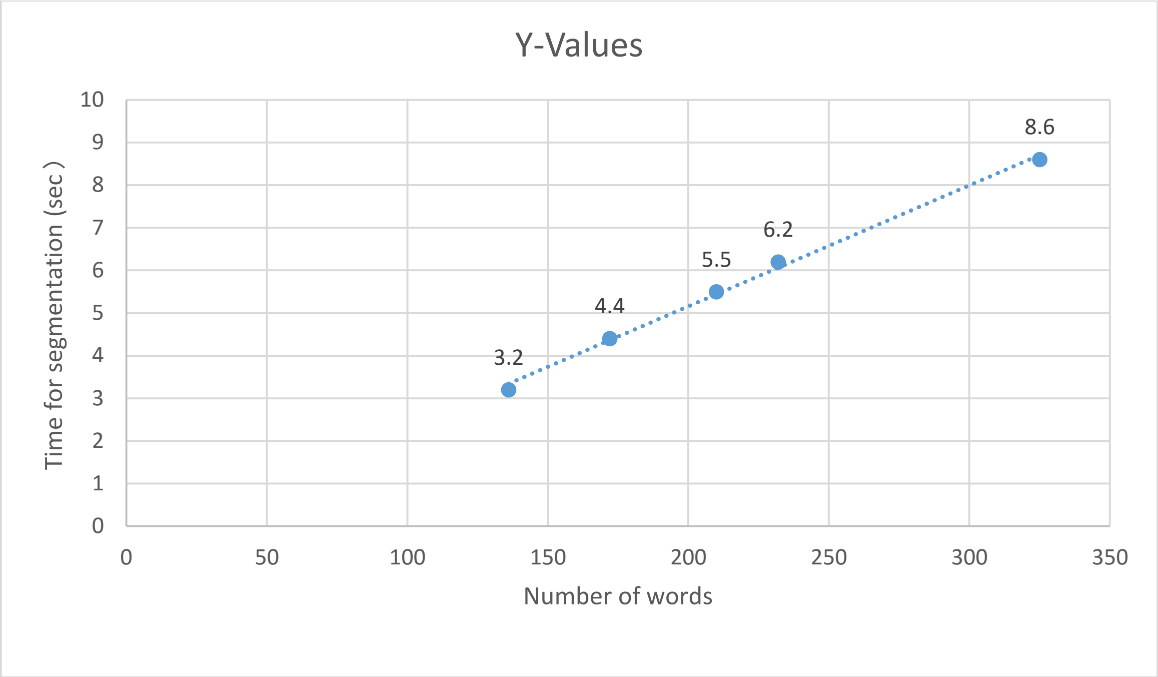


Figure 6.3(a): Number of words VS Time taken for segmentation & recognition Graph

Chapter 7
CONCLUSION

7.1 Conclusion

This paper delineates the overview of Bangla text image, especially the scanned image, from the perspective of OCR. It describes approaches for extracting features from an image and which feature is best fitted for recognition process. Here we have described one approach for recognizing the segmented characters, and post process them for recognizing the correct words. The analysis depicted in this report will be informative and helpful for the upcoming researchers who will be interested to work in this highly challenging field.

7.2 Future Work

Our implemented approach works enough well for scanned text image except some trivial constrain. We have worked on only one font “Solaiman Lipi” and have used only one feature for recognition process. In future ANN (Artificial Neural Network) can be implemented using multiple feature in different layers. Also a font independent OCR system have to be developed.

REFERENCES

1. Farjana Yeasmin Omee & Md. Shiam Shabbir Himel "DEVELOPMENT OF SHABDAYON BANGLA OCR"
2. http://www.iskysoft.com/convert-pdf/what-is-ocr.html
3. Md. AbulHasnat, S. M. Murtoza Habib, Mumit Khan, "Segmentation free Bangla OCR using HMM: Training and Recognition"
4. http://bocra.sourceforge.net/doc/ .
5. http://www.apona-bd.com/apona-pathak/bangla-ocr-apona-pathak.html.
6. http://sourceforge.net/project/showfiles.php?group_id=158301&package_id=215908.
7. http://www.unb.com.bd/print/muhith-ocr
8. MinhazFahimZibran, ArifTanvir, RajiullahShammi and Ms. AbdusSattar, Computer Representation of Bangla Characters And Sorting of Bangla Words, Proc. ICCIT‟2002 , 27-28 December, East West University, Dhaka, Bangladesh.
9. ArifBillah Al-Mahmud Abdullah and MumitKhan,"A Survey on Script Segmentation for Bangla OCR" Dept. of CSE, BRAC University, Dhaka, Bangladesh
10. Md. MahbubAlam and Dr. M. AbulKashem, "A Complete Bangla OCR System for Printed Chracters" JCIT-100707.pdf
11. Tushar Patnaik, Shalu Gupta, Deepak Arya, "Comparison of Binarization Algorithm in Indian Language OCR".
12. http://en.wikipedia.org/wiki/Otsu's_method
13. Nasreen Akter, Saima Hossain, Md. Tajul Islam & Hasan Sarwar (2008). An Algorithm For Segmenting Modifies From Bangla Text, ICCIT, IEEE, Khulna,Bangladesh, PP.177-182
14. B.B. Chaudhuri & U. Pal (1998). Complete Printed Bangla OCR System, Elsevier Science Ltd.
    Pattern Recognition, Vol(31): 531 -549
15. Saima Hossain, Nasreen Akter, Hasan Sarwar and Chowdhury Mofizur Rahman United International University Bangladesh "Development of a recognizer for Bangla text: present status and future challenges" .

16. Praveen Kumar, Shivangi garg, Sandeep Tiwari "Character Recognition using Neural Network" IMS Engineering College, Adhyatmic Nagar, Dasna, Ghaziabad, UP, India.
17. Adnan Mohammad Shoeb Shatil and Mumit Khan "Minimally Segmenting High Performance Bangla Optical Character Recognition Using Kohonen etwork" BRAC University, Dhaka, Bangladesh.
18. http://docs.opencv.org/doc/tutorials/features2d/trackingmotion/harris_detector/harris_detector.html.
19. https://sites.google.com/site/rameyarnaud/research/c/voronoi