ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

A New Method for Recognition of Handwritten Characters Using Edge Detection, Segmentation and Pattern Matching

Monish Dutta, Pritha Roy, Sunanda Datta, Asoke Nath

M.Sc. Student, Department of Computer Science, St. Xavier's College (Autonomous), Kolkata, India

M.Sc. Student, Department of Computer Science, St. Xavier's College (Autonomous), Kolkata, India

M.Sc. Student, Department of Computer Science, St. Xavier's College (Autonomous), Kolkata, India

Associate Professor, Department of Computer Science, St. Xavier's College (Autonomous), Kolkata, India

ABSTRACT: To convert handwritten character to printed character is a very challenging problem in computer science and information technology. It can save time and makes human's job easier. Character reorganization device is one of such smart devices that acquire partial human intelligence with the ability to capture and recognize various characters and digits. Character recognition comes into picture when various patterns of handwritten or printed characters are to be recognized digitally. Many researchers have proposed different approaches for character recognition in different languages. In this paper, a system is developed for the recognition of basic characters (vowels, consonants and numbers) in handwritten and printed English text, which can handle different font sizes and font types. The main important phases of character recognition include pre-processing, segmentation, feature extraction and matching. The authors proposed a new but simple method to recognize Handwritten Character. There is a scope for further enhancement of this method to make the algorithm more robust and intelligent.

KEYWORDS: Handwritten character; English text; Basic Characters; Offline Character recognition; Online character recognition

I. INTRODUCTION

Handwriting recognition is in research for over four decades and has attracted many researchers across the world. Character recognition systems translate such scanned images of printed, typewritten or handwritten documents into machine encoded text. This translated machine encoded text can be easily edited, searched and can be processed in many other ways according to requirements. Character recognition systems help humans ease and reduce their jobs of manually handling and processing of documents. Computerized processing to recognize individual character is required to convert scanned document into machine encoded form. Character recognition is mainly of two types online and offline. In online character recognition, data is captured during the writing process with the help of a special pen on electronic surface. In offline recognition, prewritten data generally written on a sheet of paper is scanned. In the present work the authors proposed an offline handwritten character recognition method.

1) Offline Character Recognition : Generally all printed or type-written characters are classified in offline mode. Off-line handwritten character recognition refers to the process of recognizing characters in a document that have been scanned from a surface such as a sheet of paper and are stored digitally in grey scale format. The storage of scanned documents have to be bulky in size and many processing applications as searching for a content, editing, maintenance are either hard or impossible. Such documents require human beings to process them manually, for example, postman's manual processing for recognition and sorting of postal addresses and zip code. Character recognition systems translate such scanned images of printed, typewritten or handwritten documents into machine encoded text. This translated machine encoded text can be easily edited, searched and can be processed in many other ways according to requirements. It also requires tinny size for storage in comparison to scanned documents.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

2) Online Character Recognition : The online mode of recognition is mostly used to recognize only handwritten characters. In this the handwriting is captured and stored in digital form via different means. Usually, a special pen is used in conjunction with an electronic surface. As the pen moves across the surface, the two- dimensional coordinates of successive points are represented as a function of time and are stored in order. Recently, due to increased use of handheld devices online handwritten recognition attracted attention of worldwide researchers. This online handwritten recognition aims to provide natural interface to users to type on screen by handwriting on a pad instead of by typing using keyboard. The online handwriting recognition has great potential to improve user and computer communication. In online handwriting recognition, it is very natural for the user to detect and correct misrecognized characters on the spot by verifying the recognition results as they appear. The user is encouraged to modify his writing style so as to improve recognition accuracy. Also, a machine can be trained to a particular user's style. Samples of his misrecognized characters are stored to aid subsequent recognition. Thus both writer adaptation and machine adaptation is possible.

II. METHODS USED IN RECOGNIZING HANDWRITTEN CHARACTERS

The approach used in this paper is summarised as below.

Firstly handwritten texts are to be scanned using digital camera or a scanner and create a digital jpeg image. The digital image is read by an inbuilt function and then takes the whole image in a 2d array form where each element is the value of each pixel. Then it performs rgb2binary operation to convert all the pixel value to 1 or 0. That means it takes the drawn image pixels as value 1 and other white pixels as 0. After the binarization operation it selects only the drawn image from the original image by eliminating white part in 4 sides using cropimage operation. Then it performs the topLineEdge operation for detecting the end row of top line of the cropped image. Then it selects only the top line from the image. The source image is set to the rest of the source image by eliminating the top line image using crop operation. After that the cropped line image is used for charEdge operation where the left most first character end edge is detected and the first character is selected by crop operation. The line image is set to the rest of the image that means by eliminating the first character. Then the first character is used for match operation. And the result of it is stored in a text file. After matching the first character image again it takes the rest of the top line image and fetches the first character of it and this operation is performed until the top line image width is 1 that is no character is remaining in the top line image. Similarly after performing all operations on top line image, the top line image from rest of the source image is selected and the same operation is performed. Each next line image is to be treated as a new line and that is inputed in a text file. After fetching each character from each line image, the rest of the line image is selected using the boundary values which is getting from Edgedetection operation to eliminate the first character image and the space between each character and that space is used to determine the blank space between words, that means if the space is greater than some predefined value it will be detected as a space between words and puts a space on the text file.

Before the matching it first resizes both the character image and database image in same resolution otherwise the match operation cannot be done. That is why it takes lcm of both the height of two images and the width of two images and resizes both the image into newly computed height and width using a inbuilt function imresize.

IMRESIZE(A, [NUMROWS NUMCOLS],'nearest') resizes the image so that it has the specified number of rows and columns. Either NUMROWS or NUMCOLS may be NaN, in which case IMRESIZE computes the number of rows or columns automatically in order to preserve the image aspect ratio. 'nearest' is used for nearest-neighbour interpolation
Following are the operations performed on the input image :

Image Acquisition: The following procedures applied on scanned handwritten character image:

Image Processing : It consists of three processes Binarization , Edge detection, and Image crop.
In *binarization* method it takes the original scanned image matrix as input and processes each pixel of it in such a way that, if the pixel value is greater than 127 it assumes it as a white pixel and change it to value 0. Similarly if the pixel value is less than 127 it assumes it as a black pixel and change it to 1.

In *Edgedetection* the boundary of input image is detected. It takes the binary image matrix as input, scan it from top row to bottom row for every pixel of each row. If it finds any pixel in a row then it assumes it as the top of image. After

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

that it again scans all pixel of every row and if it finds any pixel then it will increment the row number. At the end of image, it will save the last row where it found the pixel in the image, and assume that row as the bottom of the image. Similarly by the same process it will detect the right end and left end of that image. It will scan it from left most column to right most column for every pixel of each column. If it finds any pixel in a column then it assumes it as the left most edge of image. After that it again scans all pixel of every column and if it finds any pixel it will increment the column number. At the right most end of image it will save the last column number where it found the pixel in the image, and assume that column as the right most edge of the image.

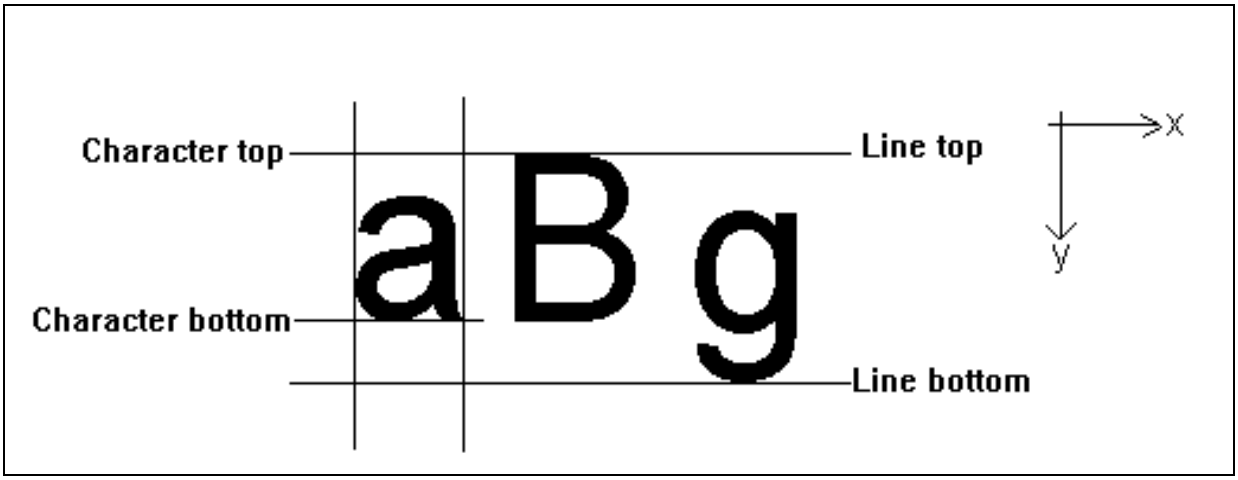


Fig 1: Line and Character boundary detection

In *cropImage* the selection operation is performed. It takes as input the image matrix boundary edges and selects only the pixel values that are within that input boundary range of the input image and save it to another matrix.

Character Segmentation : This consists of two processes Top line detection and Character separation.

In *topLineEdge* , it detects the top line of an input image and returns last row index of the top line. It first scans all pixel of each row from top. It tries to find the row scanning from top which have no pixel in any of its column. It then assumes it as the end of top line and returns the end index of that row.

In *CharEdge*, it detects the left character of an input image and returns last column index of the leftmost character. It first scans all pixel of each column from left end. It tries to find the column scanning from left which have no pixel throughout its height. It then assumes it as the end of first left most character and returns the end column index.

Character Image Matching : In this process, the matching operation of an input image with the image that is stored in the database is performed. It performs pixel by pixel matching between the input image and all the stored image in the database. The algorithm checks whether any pixel value of input image is matching with the same pixel value of database image. If it matches it increments the number of matched pixel counter by 1 and takes the percentage of it, that is the percentage of matched pixel to total number of pixels. It then do the same for all the database image and the one which have maximum match percentage is selected as the matched image and according to this appropriate character will be detected.

The following Block diagram gives the stepwise procedure for detecting characters.

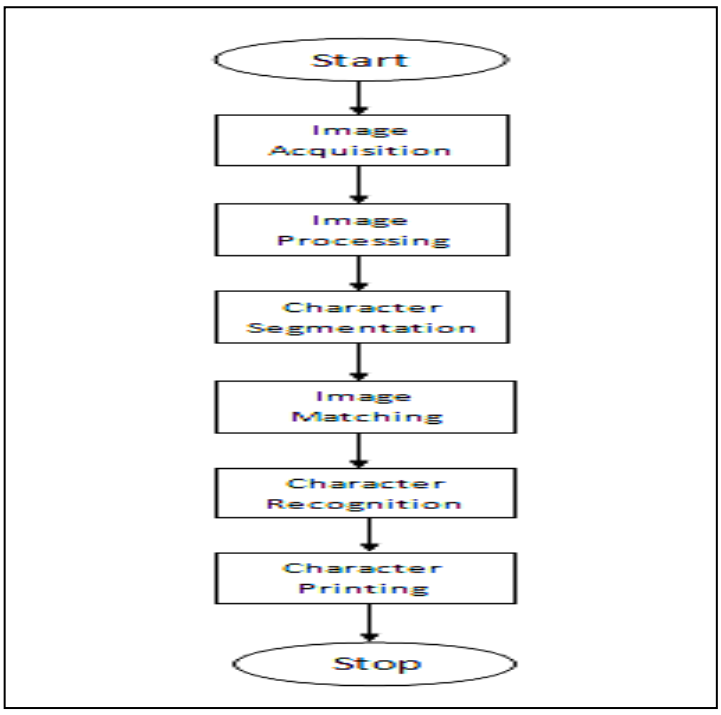
ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

Fig 2 : Stepwise Procedure

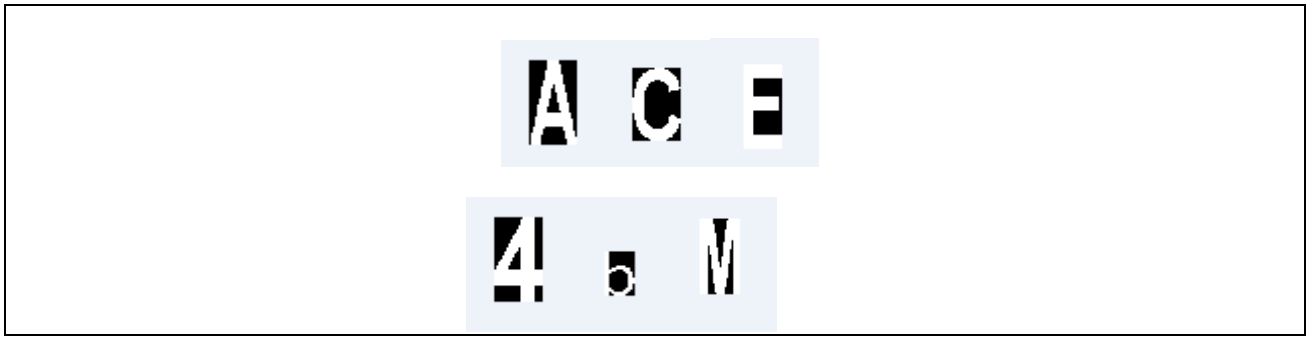


Dataset Creation

For creating the data base, first a drawn printed character image is taken and fetched the character using the same operation described above . No matching operations are performed but instead of that it saves the image matrix in .bmp file using file write operation.

Some of the database samples are shown below:

Fig 3 : Dataset Samples



ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

III. PROPOSED ALGORITHM

Algorithm to convert an image to its binary form

Procedure: rgb2binary

Step 1: Take the image matrix as input.
Step 2 : Initialize i = 1
Step 3 : Initialize j = 1
Step 4 : If any pixel value >=127 then
pixel value = 0
else
pixel value= 1
End if
Step 5 : j=j+1
Step 6 : if j<=width then go to step-
Step 7 : i=i+1
Step- 8 : if i<=height then go to step-3
Step 9 : End procedure

Algorithm for determining Top, Bottom, Left and Right edges of an image

Procedure: Edgedetection

Step 1 : Take the binary image as input.
Step 2 : Calculate the size of the image matrix.
Step 3 : Initialize flag=0
Step 4 : Initialize i = 1
Step 5 : Loop until i= height of the image do
Step 6 : Initialize j = 1
Step 7 : Loop until j = width of the image do
Step 8 : If the pixel value = 1 then
Step 9 : rowtop = i
Step 10 : flag=1
Step 11 : goto step 14
Step 12 : End if
Step 13 : End Loop
Step 14 : If flag = 1
Step 15 :goto Step 18
Step 16 : End if
Step 17 : End Loop
Step 18 : Initialize i = rowtop
Step 19 :Loop until i = height of the image do
Step 20 : Initialize j = 1
Step 21 : Loop until j = width of the image do
Step 22: If pixel value = 1 then
Step 23 : rowbottom = i
Step 24 : End if
Step 25 : End Loop
Step 26 : End Loop
Step 27 : Initialize flag=0
Step 28 : Initialize i = 1

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

Step 29 : Loop until i = width of the image do
Step 30 : Initialize j = 1
Step 31 : Loop until j = height of the image do
Step 32 : If the pixel value = 1 then
Step 33 : columnleft = i
Step 34 : flag=1
Step 35 : goto step 38
Step 36 : End if
Step 37 : End Loop
Step 38 : If flag = 1
Step 39 : goto Step 42
Step 40 : End if
Step 41 : End Loop
Step 42 : Initialize i = 1
Step 43 : Loop until i = width of the image do
Step 44 : Initialize j = 1
Step 45 : Loop until j = height of the image do
Step 46 : If pixel value = 1 then
Step 47 : columnright = i
Step 48 : End if
Step 49 : End Loop
Step 50 : End Loop
Step 51 : End of procedure

Algorithm to crop a portion from an image

Procedure: cropImage
Output : s = output matrix

Step 1: Take the edge detected image matrix as input
Step 2 : Take the rowtop,rowbottom, columnleft, column right selection as input
Step 3 : Initialize k=1
Step 4 : Initialize l=1
Step 5 : Initialize i= rowtop
Step 6 : Loop until i=rowbottom of the image do
Step 7 : Initialize j= columnleft
Step 8 : Loop untilj=columnright of the image do
Step 9 : If pixel value = 1 then
Step 10 : store 1 in output matrix s(k,l)
Step 11 : Else
Step 12 : store 0 in output matrixs(k,l)
Step 13: end if
Step 14 : l=l+1
Step 15 : end loop
Step 16 : k=k+1
Step 17 : l=1
Step 18 : end loop
Step 19 : end procedure

Algorithm for selecting each line from image

Procedure: topLineEdge

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

Step 1: Take the cropped image matrix as input
Step 2 : Calculate the size of the image
Step 3 : Initialize r = 0
Step 4 : Initialize i = 1
Step 5 : Loop until i= height of the image do
Step 6 : Initialize flag= 0
Step 7 : Initialize j = 1
Step 8 : Loop until j = width of the image do
Step 9 : if the pixel value = 1 then
Step 10 : set flag =1
Step 11 : Goto step 16
Step 12 : Else
Step 13 : Flag=2
Step 14 : End if
Step 15 : End loop
Step 16 : If m = 2 then
Step 17 : r = i-1
Step 18 : goto step 21
Step 19 : End if
Step 20 : End loop
Step 21 : End procedure

Algorithm for selecting each character from a line

Procedure: CharEdge

Step 1: Take the cropped image matrix as input
Step 2 : Calculate the size of the image
Step 3 : Initialize x = 0
Step 4 : Initialize i = 1
Step 5 : Loop until i= width of the image do
Step 6 : Initialize flag= 0
Step 7 : Initialize j = 1
Step 8 : Loop until j =height of the image do
Step 9 : if the pixel value = 1 then
Step 10 : set flag =1
Step 11 : Goto step 16
Step 12 : Else
Step 13 : Flag=2
Step 14 : End if
Step 15 : End loop
Step 16 : If m = 2 then
Step 17 : x = i-1
Step 18 : goto step 21
Step 19 : End if
Step 20 : End loop
Step 21 : End procedure

Algorithm for Matching each character

Input : a = Character image
m = Database image

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

Procedure:Match

Step 1 : Take each character image and database image as input and repeat the procedure for all the images.
Step 2 : Calculate the size of each character image.
Step 3 : Calculate the size of each database image.
Step 4 : Initialize the counter c = 0 for matching.
Step 5 : Take the LCM of the rows of character image and the database image.
Step 6 : Take the LCM of the columns of character image and the database image.
Step 7 : Resize the character image by taking the row and column value from Step 5 and Step 6 respectively.
Step 8 : Resize the character image by taking the row and column value from Step 5 and Step 6 respectively.
Step 9 : Initialize i = 1.
Step 10 : Loop until i = row do
Step 11 : Initialize j = 1.
Step 12 : Loop until j = column do
Step 13 : if a(i,j) = m(i,j) then
Step 14 : c = c+1
Step 15 : End if
Step 16 : End Loop
Step 17 : End Loop
Step 18 : Calculate the matching percentage c/(r*cl)*100.
Step 19 : if c < 50
Step 20 : Return -1
Step 21 : else
Step 22 : Return c
Step 23 : End if
Step 24 : End procedure

Algorithm main
Variables : rt = rowtop
rb = rowbottom
cl = columnleft
cr = columnright

Step 1 : Take image file as input.
Step 2 : Read the image file and store it in p.
Step 3 : Calculate the size of the image.
Step 4 : Convert the image to binary and again store it in p.
Step 5 : Call edgeDetection function on p for detecting rt , rb , cl , cr.
Step 6 : Call cropImage function on p and store the modified image matrix in a.
Step 7 : Calculate size of the cropped image a and store the height in r1 and width in c1.
Step 8 : Store r1 in r2, r1=r2.
Step 9 : Store c1 in c3, c3=c1.
Step 10 : While (height of the cropped image a) r2 > 1
Step 11 : Call topLineEdge to calculate the bottom of top line of cropped image a and store in r
Step 12 : If r = 0 then
Step 13 : Goto step 60
Step 14 : End if
Step 15 : Call cropImage on image matrix a and store it in b.
Step 16 : Call edgeDetection for b and store it in rt2 , rb2 , cl2 , cr2 .
Step 17 : Call cropImage for b and again store it in b.
Step 18 : Calculate size of b and store the height in r4 and width in c4.
Step 19 : Store c4 in c3, c3=c4 .
Step 20 : While (width of the cropped image b) c3> 1 .

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

Step 21 : Call charEdge on image matrix b to calculate the columnright of the image and store in x.
Step 22 : If x = 0 then
Step 23 : Goto step 47
Step 24 : End if
Step 25 : Call cropImage on image matrix b and store it in d.
Step 26 : Call edgeDetection for d and store it in rt3 , rb3 , cl3 , cr3.
Step 27 : Call cropImage on b and store it in d.
Step 28 : Calculate size of d and store the height in r5 and width in c5.
Step 29 : Open a text file.
Step 30 : Read the character in d and store it in l.
Step 31 : Print l in the text file.
Step 32 : If width of cropped image b – columnright<=1 then
Step 33 : Goto step 47
Step 34 : End if
Step 35 : Call cropImage for b and again store it in b.
Step 36 : Call edgeDetection for b and store it in rt2 , rb2 , cl2 , cr2.
Step 37 : If the (columnLeft) cl2 > 7
Step 38 : Print space
Step 39 : End if
Step 40 : If c3 = 1 then
Step 41 : Goto step 47
Step 42 : End if
Step 43 : Call cropImage for b and again store it in b.
Step 44 : Calculate size of b and store the height in r4 and width in c4.
Step 45 : Store c4 in c3, c3=c4.
Step 46 : End while
Step 47 : If r2-r < =1then
Step 48 : Goto step 60
Step 49 : End if
Step 50 : Call cropImage on a and again store it in a.
Step 51 : Call edgeDetection on a and store it in rt1, rb1, cl1, cr1.
Step 52 : If r2 = 1 then
Step 53 : Goto step 60
Step 54 : End if
Step 55 : Call cropImage for a and again store it in a.
Step 56 : Print next line.
Step 57 : Calculate size of a and store height in r2 and width in c2.
Step 58 : Store c2 in c1, c1=c2.
Step 59 : End while
Step 60 : Close the text file.
Step 61 : End procedure

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

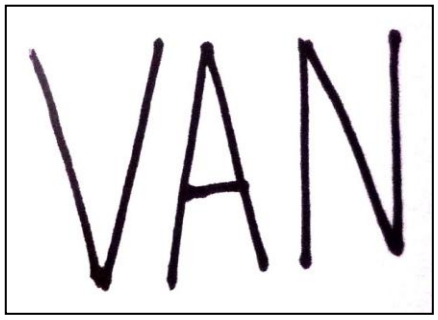
(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

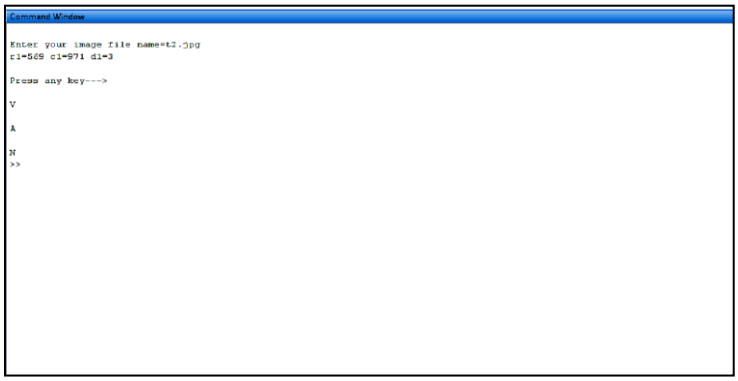
IV. RESULTS AND DISCUSSION

SET 1:

Input Image:

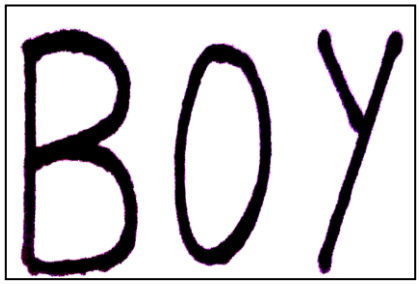


Output Image:

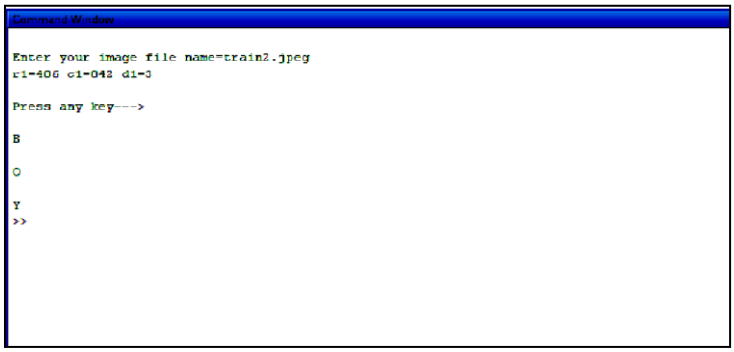


SET 2:

Input Image:



Output Image



SET 3:

Input Image:

Output Image

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

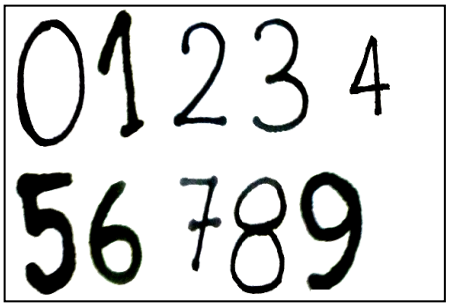
Vol. 3, Issue 6, June 2015

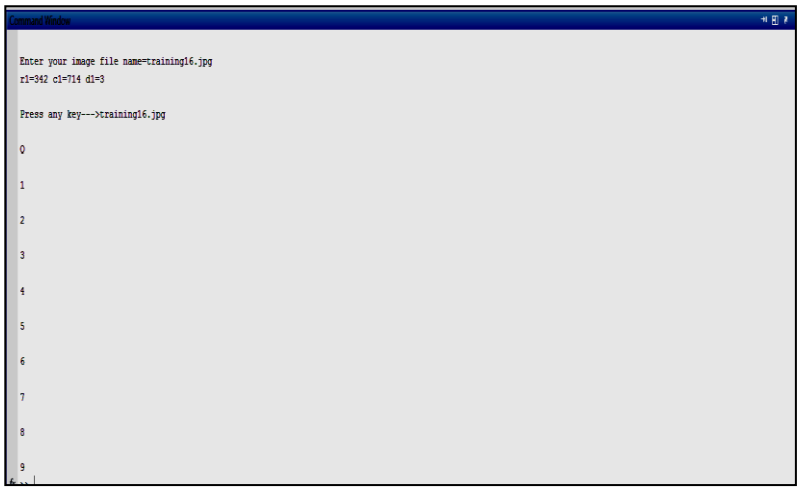
SET 4:

Input Image: Output Image

SET 5:

Input Image: Output Image





V. LIMITATIONS OF THE PRESENT METHOD

The present method has the following limitations:
(i) It performs the matching operation pixel by pixel basis so only simple handwritten character that looks like printed character can give good result.
(ii) The processing time is comparatively slow.
(iii) The present work does not detect small characters.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

VI. CONCLUSIONS AND FUTURE SCOPE

In the present paper handwritten character recognition is categorized into three main divisions: Image Acquisition, Image processing and Matching with the dataset. The experimental results illustrate how an input image leads us to character extraction and recognizes the handwritten characters accordingly. The designed systems have the ability to yield accurate results, provided the correct dataset is available. There is a scope of improvement in the current system using Neural Network. Hence, a simple yet effective approach for recognition of handwritten characters has been described. The recognition accuracy of the prototype implementation is promising, but more work needs to be done. In particular, no fine-tuning of the system has been done so far. Our character segmentation method also needs to be improved so that it can handle a larger variety of touching characters, which occur fairly often in images obtained from inferior-quality printed material. In general, the system needs to be tested on a wider variety of images containing characters in diverse fonts and sizes. This will enable us to identify the major weaknesses in the system and implement remedies for them. This program may detect small letter characters and special characters with the proper creation of dataset.

REFERENCES

1. Anshul gupta and Manisha srivastava , "offline handwritten character recognition" , Department of electronics and communication engineering, Indian Institute of Technology ,Guwahati, Assam, India – 781039, April, 2011.
2. Neeta Nain Subhash Panwar ,"Handwritten Text Recognition System Based on Neural Network", Malaviya National Institute of Technology Jaipur, Neeta Nain Subhash Panwar.
3. Vinita Dutt1, Sunil Dutt, "Hand Written Character Recognition Using Artificial Neural Network", Advances in Computing: 2011; 1(1): 18-23 DOI: 10.5923/j.ac.20110101.03.
4. Hassoun, M.H., "Fundamentals of Artificial Neural Networks", MIT Press, Cambridge, London, England,1995.
5. Avi Drissman, Dr. Sethi, CSC 496 ,February 26, 1997, "Handwriting Recognition Systems: An Overview".
6. MANSI SHAH AND GORDHAN B JETHAVA**,** Department of Computer Science & Engineering Parul Institute of Technology, Gujarat, India.Information Technology Department Parul Institute of Engg. & Technology, Gujarat, India, "A LITERATURE REVIEW ON HAND WRITTEN CHARACTER RECOGNITION".
7. Suruchi G. Dedgaonkar, Anjali A. Chandavale, Ashok M. Sapkal , "Survey of Methods for Character Recognition"
8. Sukhpreet Singh, M.tech Student, Dept. of Computer Engineering, YCOE Talwandi Sabo BP. India, "Optical Character Recognition Techniques: A Survey".
9. Priya Sharma, Randhir Singh,Lecturer, Department of ECE, SAI Polytechnic, Badhani, Punjab, India Head, Department of ECE, SSCET, Badhani Punjab, India, International Journal of Engineering Trends and Technology- Volume4Issue3- 2013," Survey and Classification of Character Recognition System".
10. Jagruti Chandarana1, Mayank Kapadia2 ,1Department of Electronics and Communication Engineering, UKA TARSADIA University ,Assistant Professor, Department of Electronics and Communication Engineering, UKA TARSADIA University,International Journal of Emerging Technology and Advanced Engineering Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 4, Issue 5, May 2014)," Optical Character Recognition".

BIOGRAPHY

Monish Kumar Dutta is a stdunet of M.Sc, Department of Computer Science, St. Xavier's College(Autonomous), Kolkata.

Pritha Roy passed M.Sc in Computer Science in 2015 from, Department of Computer Science, St. Xavier's College(Autonomous), Kolkata.

Sunanda Datta passed M.Sc in Computer Science in 2015 from, Department of Computer Science, St. Xavier's College(Autonomous), Kolkata.

Dr. Asoke Nath is Associate Professor in Department of Computer Science, St. Xavier's College(Autonomous), Kolkata. He is involved in research areas like Cryptography and Network security, Visual Cryptography, Steganography, Green Computing, MOOCs and e-learning, Social Networks, Big data etc. He has published more than 145 research papers in International and National Journals and Conference Proceedings.