TRANSFORMING PRINTED DOCUMENTS INTO   
DATABASE   
   
   
A PROJECT REPORT   
   
Submitted by   
   
NISARG N PATEL   
POOJA M PATEL   
POOJA H PATEL   
   
   
In partial fulfillment for the award of the degree   
of   
   
BACHELOR OF ENGINEERING   
   
in   
   
COMPUTER ENGINEERING   
   
   
   
   
   
SARDAR VALLABHBHAI PATEL INSTITUTE OF TECHNOLOGY   
VASAD   
   
   
Gujarat Technological University, Ahmedabad   
April, 2013  
   
   
   
   
   
   
SARDAR VALLABHBHAI PATEL INSTITUTE OF TECHNOLOGY   
   
COMPUTER ENGINEERING   
   
2013   
   
   
   
   
CERTIFICATE   
   
Date:30 April 2013   
   
This is to certify that the PROJECT entitled “TRANSFORMING PRINTED   
DOCUMENTS TO DATABASE” has been carried out by NISARG N PATEL   
(090410107081), POOJA M PATEL (090410107086), POOJA H PATEL (090410107090)   
under my guidance in partial fulfillment of the degree of Bachelor of Engineering in   
COMPUTER ENGINEERING (7th & 8th Semester) of Gujarat Technological University,   
Ahmedabad during the academic year 2012-13.   
   
   
   
   
.   
Project Guide :   
   
   
   
   
   
   
Head Of Department:   
   
   
   
   
   
   
Mr. Rashmin Prajapati,   
   
   
   
   
   
Prof. B.J Talati   
Assistant Professor,   
   
   
   
   
   
   
H.O.D,   
CE Department,   
   
   
   
   
   
   
CE Department,   
S.V.I.T Vasad.   
   
   
   
   
   
   
S.V.I.T Vasad.   
   
   
   
   
   
   
   
   
   
   
ACKNOWLEDGEMENT   
We students of B.E, Sardar Vallabhbhai Patel Institute Of Technology of Computer   
Application hear by express our thanks to Developers for giving us to do the project on   
Transforming Handwritten and Printed documents to database tables. This project work has   
been the most exciting part of our learning experience, which would be an asset for our future   
carrier.   
We would like to express our sincere gratitude to Mr. Viral Patel for his guidance and   
constant inspiration with the valuable suggestions during our project work for providing us all   
the necessary information for designing and developing the project.   
We are also indebted to for him encouragement and exclusive help, without which we would   
have been lacking something. Knowledge in itself is a continuous process getting practical   
knowledge is important thing which is not possible without the support, guidance, motivation   
and inspiration provided by different persons.   
We are also greatly thankful to BE staffs that have helped us in completion of this project   
directly or indirectly throughout our academic semester and for encouraging us to take all the   
facilities.   
Moreover we would also like to thank our friends and last we are grateful to our parents for   
their support and unconditional help, which made our project a real success.   
Specially thanks to,   
PROJECT GUIDE:   
 Mr. Viral Patel   
 Mr. Rashmin Prajapati   
   
   
   
   
   
   
   
ABSTRACT   
   
Every Educational Institutes need some kind of formatted marksheet. Here in   
our project we make work simpler for this institutes.   
   
Transforming printed or handwritten documents directly to database. For   
example we take marksheet of GTU, once it is distributed to Institutes, the   
Institutes need to maintain the records, so for reducing the manually work load   
this software is usefull.   
   
In this software the user just need to scan the copy of marksheet and rest of the   
thing is done by software. The scanned file is stored as Image file so this Image   
file undergoes processing and the useful data is extracted. This data is stored   
into database , thus reducing the manual burden . Our project works for any   
kind of format, so this makes our project Dynamic.   
   
Thus our project has to undergo various Image Processing task. This makes   
things simpler for the Educational Institutes.   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
LIST OF TABLES   
   
   
Table No Table Description   
Page No   
   
Table 4.3.1   
MASTER\_TEMPLATE\_TABLE   
   
   
   
   
39   
   
Table 4.3.2   
CHILD\_TEMPLATE\_TABLE   
   
   
   
   
40   
   
Table 4.3.3   
VALUE\_TABLE   
   
   
   
   
   
   
41   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
LIST OF FIGURES   
   
   
   
   
Figure No Figure Description   
Page No   
   
4.2   
   
USE CASE DIAGRAM   
   
   
   
   
   
35   
   
   
3.1   
   
BLOCK DIAGRAM   
   
   
   
   
   
10   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
LIST OF SYMBOLS, ABBREVIATIONS AND NOMENCLATURE   
   
   
   
Symbol   
Abbreviations   
   
Name   
   
OCR   
Optical Character Recognition   
SQL   
Structured Query Language   
DPI   
   
Dots per Inch   
JPEG   
Join Photographic Expert Group   
BMP   
Bitmap Image File Format   
TIF   
   
Tagged Image File Format   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
TABLE OF CONTENTS   
   
   
   
Acknowledgement   
i   
   
   
Abstract   
ii   
   
List of Figures   
iii   
   
List of Tables   
iv   
   
List of Abbreviations   
v   
   
Table of Contents   
vi   
Chapter : 1   
INTRODUCTION   
   
   
   
1.1 Document Purpose   
1   
   
   
1.2 Project Scope   
1   
   
   
 1.2.1 Application   
1   
   
   
 1.2.2 Goal   
1   
   
   
1.3 Overview   
2   
   
   
1.4 Project Profile   
2   
   
Chapter : 2   
PROJECT DESCRIPTION   
   
   
   
2.1 Product Perspective   
3   
   
   
 2.1.1 System Interface   
3   
   
   
 2.1.2 User Interface   
3   
   
   
 2.1.3 Hardware Interface   
   
 8   
   
   
 2.1.4 Software Interface   
   
   
   
 8   
   
   
 2.1.5 Communication Interface   
   
   
   
 8   
   
   
 2.1.6 Memory Constrains   
8   
   
   
2.1.7 Operations   
8   
   
   
2.1.8 Site Adaptation Requirements   
8   
   
   
 2.2 Product Functions   
8   
   
Chapter : 3   
LITERATURE SURVEY   
   
   
   
3.1 Processing Steps   
10   
   
   
 3.1.1 Pre-Processing   
10   
   
   
 3.1.2 Segmentation   
15   
   
   
   
   
   
   
 3.1.3 Character Recognition   
18   
   
   
 3.2 Cropping an Image   
 31   
   
   
   
   
   
 Chapter : 4   
SYSTEM SPECIFICATION   
 4.1 Specification 33   
 4.1.1 Input Specification 33   
 4.1.2 Output Specification   
   
   
   
 33   
 4.1.3 Functional Specification 33   
   
   
 4.2 Actor Definition   
   
   
 35   
   
   
   
 4.2.1 Junior Clerk   
   
   
   
   
   
 36   
   
   
   
 4.2.2 Head Clerk   
   
   
   
   
   
 36   
   
   
   
 4.2.3 Image Input   
   
   
   
   
 36   
   
   
   
 4.2.4 Image Processing   
   
   
   
   
 36   
   
   
   
 4.2.5 Save   
   
   
   
   
   
 37   
   
   
   
 4.2.6 Edit   
   
   
   
   
 37   
   
   
   
 4.2.7 Pre-Processing   
   
   
   
   
 37   
   
   
   
 4.2.8 Segmentation   
   
 37   
   
   
   
 4.2.9 Character Recognition   
   
   
   
   
 38   
   
   
   
 4.2.10 Table And Template Definition   
   
   
 38   
   
   
 4.3 Tables   
   
   
   
   
   
   
 39   
   
Chapter : 5   
RESULT   
   
   
   
   
 42   
   
Chapter : 6   
CONCLUSION   
   
   
   
   
   
 45   
   
   
 6.1 Conclusion   
   
   
   
   
   
   
 45   
   
   
 6.2 Scope Of Future Enhancement   
   
   
   
   
 45   
   
Chapter : 7 REFRENCES   
   
   
   
   
   
   
 46   
   
   
   
   
   
   
   
   
   
   
   
 CHAPTER-1 INTRODUCTION   
1.1 Document Purpose   
This Software Requirements Specification provides a complete description of all the functions   
and specifications of the OCR for Printed and Hand written documents. This documentation   
presents an intense study of requirements of OCR system where we can scan hardcopy of   
documents in scanner and store its data in database after processing thus reducing the manual   
burden of entering the data in database .   
1.2 Project Scope   
The system transformations printed and hand written text on any kind of form/format into the   
database. The text could be on plain paper or else on pre-printed form. The database design   
including normalization is manually done. The scanned document, which will be in the image   
format, will be the input. This scanned image undergoes pre-processing and data will be   
stored in database.   
   
   
1.2.1 Application   
 Application areas of this system are very large as in many organizations, for eg:   
   
E-Governance   
   
Government organizations   
   
Administrative Offices   
   
Off-line competitive examination management system   
   
Many statutory forms/results prepared in various organization   
   
1.2.2 Goal   
Goal of this project is to develop a system that focuses on following major criteria:   
 Record keeping and archiving   
 Efficient and automated record storage and indexing   
 Increased accuracy   
 Reduced Time   
 Reduced Manual Burden   
 To recognize Hand written character (for Vernacular language fonts- Gujarati   
Language)   
 Make data available for long time without any loss.   
   
   
   
   
   
1.3   
Overview   
This SRS document details OCR system and the requirements segregated. It provides an   
introduction to this document and also provides a full description of the project along with a   
detailed list of requirements for the user of the OCR system. It lists all the functions   
performed by the system. It concerns the details of the requirement model for the OCR for   
hand written or printed documents.   
   
1.4 Project Profile   
   
   
   
   
   
Project Title   
Transformation of Printed and Handwritten document   
into the Database   
Tables   
Project Type   
OCR Based Application   
Objective   
   
   
To develop a OCR Based Application   
 Tools & Technology   
NetBeans 7.0 (java) and Matlab(Testing Tool)   
Back-End Tools   
Microsoft SQL Server – 2005   
Education Institute   
Sardar vallabhbhai Patel Institute Of Technology,   
Vasad(SVIT)   
 Project Duration   
   
1year(2012-2013)   
Organization   
   
   
   
Akshar Software Solutions,   
Karelibaug, Vadodara -22   
   
External Project Guide   
Mr. Viral Patel   
Internal Project Guide   
   
Mr. Rashmin Prajapati   
   
   
   
   
   
CHAPTER-2 PROJECT DESCRIPTION   
Project Description   
OCR is a complex technology that converts images with text into editable formats. OCR   
allows you to process scanned books, screenshots and photos with text and get editable   
documents like TXT, DOC or PDF files. This technology is widely used in many areas and   
the most advanced OCR systems can handle almost all types of images, even such complex as   
scanned magazine pages with images and columns or photos from a mobile phone.   
2.1 Product Perspective   
The figure depicts the overall system architecture for the OCR System.   
2.1.1 System Interfaces: - The system is dependent on system interfaces such as   
(a) Minimum System configuration: 3 GHz Processor, 1GB RAM, 80GB Hard   
 Disk.   
(b) Keyboard, Monitor, Mouse and Flat Bed Scanner.   
   
 2.1.2 User Interfaces :-   
Our project uses a GUI, firstly we have Login Page where the user enters the   
User Name and Password. The User Name and Password are verified from   
the Database and then only the user is allowed to use the software.   
   
   
   
   
   
   
Then the next is Home Page where the user is given the choice to whether to enter data   
into existing Template or want to make a new template.   
   
   
Once the user Click on SAVE TEMPLATE the next page will ask the user to get the   
sample Image of the template to be made.   
   
   
   
   
   
   
Once the user click on Color TO Grey Scale button image which is browsed will   
convert RGB to Grey.   
   
And then after converting to grey image user clicks on BINARIZE button grey image   
is binarize.   
   
   
   
   
   
   
   
   
   
User will enter Template Name and Template Id will be generated dynamically.   
When user clicks on NEXT button then new page is arrived in which user select the   
field that is required and according to that length ,breadth,X-coordinate and Y-  
Coordinate dynamically.   
   
   
When user clicks NEXT button ,next required field is being selected.   
   
   
   
   
Then when the user picks some item from Select Template in which image of the   
object having same template is taken, processing is done on that image and the data so   
obtained is stored in data base.   
   
   
Softwre crop the image which is required automatically.   
   
   
   
   
   
   
2.1.3 Hardware Interface:-   
(a) Monitor screen- The software shall display information to the user   
via the monitor screen.   
(b) Mouse- The software shall interact with the movement of the mouse   
and the mouse buttons. The mouse shall activate input and output,   
command buttons and select options from menus.   
(c) Keyboard- The software shall interact with the keystrokes of the   
keyboard. The keyboard will input data into the active areas of the GUI.   
(d) Flat-Bed Scanner- The software should sense the input from the   
scanner and provide the user to save the scanned document in a user   
specified format.   
   
2.1.4 Software Interfaces:-   
(a) Data Base   
(b) Operating System- Microsoft Windows XP   
   
2.1.5 Communications Interfaces: - N.A.   
   
2.1.6 Memory Constraints: - Minimum ram requirement 256MB.   
   
2.1.7 Operations: - NIL   
   
2.1.8 Site Adaptation Requirements :- NIL   
   
2.2 Product Functions   
The proposed OCR system will support the following functionalities:   
1. Color to Grey Scale Conversion   
 We have several algorithms that convert color images to black and white:   
a) Averaging   
   
   
   
   
b) Luminosity   
c) Desaturation   
d) Minimal and maximal decomposition   
e) The “single color channel” method   
f) Java built in method   
   
2. Detecting skew and corrected.   
3. Image Binarization.   
4. Text Direction Recognition.   
5. Image Cropping.   
6. Page Segmentation and Layout Analysis.   
7. Line and Word segmentation using script-independent and script-dependent features.   
Determining the contour or boundary of the letter.   
a) 4-connectivity   
b) 8-connectivity   
c) Scan line algorithm   
   
8. Symbol and text recognition.   
a) Contour Analysis Algorithm   
b) Neural Network   
9. Editing documents and conversion to alternate document   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
CHAPTER-3 LITERATURE SURVEY   
   
3.1 Processing Steps   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
 Scanner   
Preprocessin  
g and Noise   
Removal   
 Page Layout   
Analysis   
   
Text/Non Text   
Separation And   
Representation   
   
Paragraph Line   
and word level   
Representation   
 Segmentation   
   
Feature   
Extraction   
 Classification   
 Post   
Processor   
 Layout Restoration   
 Hand Presentation   
Manager   
   
   
Database   
OCR   
read   
Docume  
nt   
   
   
   
   
3.1.1 Pre-Processing   
   
This is the step taken before the major image processing task. The problem here is to   
perform some basic tasks in order to render the resulting image more suitable for the   
job to follow. In this case it may involve enhancing the contrast, removing noise, or   
identifying regions likely to contain the postcode.   
   
   
3.1.1.1 Color to Grey Scale   
 We have several algorithms that convert color images to black and white:   
A.   
Averaging   
B.   
Luminosity   
C.   
Desaturation   
D.   
Minimal and maximal decomposition   
E.   
The “single color channel” method   
F.   
Java built in method   
   
A. Averaging   
You take all the color components (R, G and B) and divide them by three.   
   
While the algorithm itself is pretty simple, it does a very good job, but it still   
has some downsides (it does a poor job representing how humans view   
luminosity).   
First we create a new image with the same height, width and attributes. Then   
we create a lookup table. The maximum value of the sum of three pixels can be   
765 (because the maximum value of each pixel can be 255). So we create a   
table of summed values, and at the i-th position we put our averaged value (this   
gets us less computational time):   
   
So if the images is 500×500, that means we do 250 000 less computations.   
B. Luminosity   
The second method relies on calculating the values based on luminosity. The   
luminosity method is a more sophisticated version of the average method. It   
also averages the values, but it forms a weighted average to account for human   
perception. We‟re more sensitive to green than other colors, so green is   
weighted most heavily. There are various formulas for calculating the new   
pixel values (our algorithm uses the first one, but you can also use any other   
formula; the last one is used by Photoshop):   
   
   
   
   
   
   
   
   
So what we do, we multiply the red, green and blue pixel values with a number   
and sum them up.   
C. Desaturation   
Desaturating an image takes advantage of the ability to treat the RGB   
colorspace as a 3-dimensional cube. Desaturation approximates a luminance   
value for each pixel by choosing a corresponding point on the neutral axis of   
the cube. Now the calculations are a bit “hard”, but I‟ve found a simpler   
method that says, that a pixel can be desaturated by finding the midpoint   
between the maximum of RGB and the minimum of RGB 2, if you want more   
detailed calculations use the method in 4.   
   
So we only take the minimum of the RGB values and the maximum of the   
RGB values and divide them by two. We‟ll create a lookup table for the   
division part, so that we‟ll have less values; theoretically the biggest minimum   
element we can have is 255 (if all the values are the same), so we must create a   
lookup table for 511 values (255 is minimum, 255 is maximum, the sum is 510,   
if we add 0 we have 511). We‟ll also need to write the min and max method,   
but that‟s simple as pie.   
D. Minimal and maximal decomposition   
Decomposition takes the highest or the lowest pixel of the RGB channel and   
sets that value. The maximal decomposition produces bright black and white   
images while minimal produces darker ones. The algorithm is really pretty   
simple as we only call the methods for calculating the min and max values   
from the previous method.   
Maximal decomposition:   
   
Minimal decomposition:   
   
   
   
   
   
   
E. The “single color channel” method   
This is also one of the simplest methods to produce a black and white image,   
and interesting enough, our cameras use it because it uses the least resources.   
For our output image we only set the values from a certain color, for instance if   
we choose red, the outputted values would be the red values of the pixel. So   
we‟ll create a method that accepts an extra int value (0 for R, 1 for G and 2 for   
B).   
F. Java built in method   
If you‟re not even remotely interested how color to grayscale conversion   
works, Java has a built in function that outputs a grayscaled image:   
   
   
   
   
   
   
3.1.1.2 Skew Removal   
A. Algorithm for finding out skew angle:   
   
1. Let the line AA' be a horizontal scanning line that starts from the top of the   
image and proceeds to the bottom of the image.   
2. Store the co-ordinates of the first black pixels encountered. Ideally, this is   
the tip of a skewed maatraa. (There may be exceptions to this. It may be a part   
of an alphabet that rises above a maatraa, in which case the angle returned will   
be false.)   
3. If the x co-ordinate of the point thus found is >width/2 , we assume that the   
page is tilted towards the right, otherwise left. To check of the page is un-  
skewed, we drop projections from the top of the page to the top most maatraa.   
If the height of any two projections are found to be same, the page is straight   
already and no de-skewing is required.   
4. If the page is tilted, we proceed to find the angle of tilt. We try to find an end   
of the top most (or any) maatraa. Then we find the angle of tilt of the maatraa   
   
   
   
   
as shown in Fig x. We then eliminate wild values (tilt>10 degrees) and keep   
averaging the values found.   
5. We use tan inverse to find the angle of tilt and then return the value.   
   
B. Algorithm for de-skewing the image:   
   
1. Using the dimensions of the skewed image, and the angle of tilt, derive the   
dimensions of the new image using these relations:   
float Point1x=(srcheight\*sine);   
float Point1y=(srcheight\*cosine);   
float Point2x=(srcwidth\*cosine-srcheight\*sine);   
float Point2y=(srcheight\*cosine+srcwidth\*sine);   
float Point3x=(srcwidth\*cosine);   
float Point3y=(srcwidth\*sine);   
float minx=min(0,min(Point1x,min(Point2x,Point3x)));   
float miny=min(0,min(Point1y,min(Point2y,Point3y)));   
float maxx=max(Point1x,max(Point2x,Point3x));   
float maxy=max(Point1y,max(Point2y,Point3y));   
int DestWidth=(int)ceil(fabs(maxx)-minx);   
int DestHeight=(int)ceil(fabs(maxy)-miny);   
Here Point 0,1,2,3 are the 4 corners of the source image.   
   
2. Use the following relations to create the new image:   
int Srcx=(int)((x+minx)\*cosine+(y+miny)\*sine);   
   
int Srcy=(int)((y+miny)\*cosine-(x+minx)\*sine);   
Where x and y are the pixel co-ordinates of the new image, and SRCx and   
SRCy are the coordinates of the source image.   
   
   
   
   
   
   
   
   
   
   
   
3.1.2 Segmentation   
Algorithms used   
A. 4-connectivity   
B. 8-connectivity   
C. Scan Line   
   
A. 4 – Connectivity   
   
   
Here in 4 – connectivity algorithm the four neighbours of the pixel are checked. Let us   
take some arbitrary pixel in beginning. Then consider its 4 neighbours. If one of the   
neighbouring pixel is of different color then the starting pixel then that pixel is stored   
into array as that pixel is a boundary pixel. As we have binarized the image the image   
will be in two colors. The text will be in black and background in white or vice versa.   
So as soon as the color of the neighbouring pixel changes that pixel is considered as   
the boundary pixel. 4 – Connectivity follow recursion so the same thing is applied to   
any of its neighbouring pixel. First it will check for the neighbouring RIGHT pixel   
then for LEFT, TOP and BOTTOM. Thus this is how we obtain the boundary points   
using this algorithm.   
   
Disadvantage   
 If the image size is too big then due to recursion memory problems occur   
 We cannot keep any limit or any loop to HAULT the algorithm.   
   
   
   
   
   
   
   
   
   
B. 8 – Connectivity   
   
   
   
Here in 8 – connectivity algorithm the eight neighbours of the pixel are checked. Let   
us take some arbitrary pixel in beginning. Then consider its 8 neighbours. If one of the   
neighbouring pixel is of different color then the starting pixel then that pixel is stored   
into array as that pixel is a boundary pixel. As we have binarized the image the image   
will be in two colors. The text will be in black and background in white or vice versa.   
So as soon as the color of the neighbouring pixel changes that pixel is considered as   
the boundary pixel. 8 – connectivity follow recursion so the same thing is applied to   
any of its neighbouring pixel. First it will check for the neighbouring RIGHT pixel   
then for LEFT, TOP, BOTTOM, TOP RIGHT, TOP LEFT, BOTTOM RIGHT and   
BOTTOM LEFT. Thus this is how we obtain the boundary points using this algorithm.   
Disadvantage   
 If the image size is too big then due to recursion memory problems occur   
 We cannot keep any limit or any loop to HAULT the algorithm.   
 Too much of recursion as 8 neighbors of the pixels are to be checked   
   
   
   
   
   
   
   
   
   
   
   
   
   
C. Scan Line   
   
   
   
   
Here we have customized the Scan Line algorithm. Scan line algorithm is used to   
detect the edges of the polygon. Now to detect the boundary of the letter we scan all   
the pixels horizontally first then vertically. While scanning the pixels horizontally if   
the next pixel to the current pixel is of different color than current pixel then the   
   
   
   
   
current pixel is the boundary pixel of the letter. As we have binarized the image the   
whole image will be converted to only to colors i.e black and white. So detecting the   
boundary by using color difference is simpler. Scanning the image horizontally will   
only give left and the right boundary points. So we need to scan the image vertically to   
get the top and the bottom boundary points. Now next we scan the image vertically.   
While scanning the pixels vertically if the next pixel to the current pixel is of different   
color than current pixel then the current pixel is the boundary pixel of the letter. So   
we get the top and bottom boundary pixels by scanning the image vertically. Thus by   
scanning the image vertically and horizontally we get all the boundary pixels of the   
image.   
   
Advantages   
 This algorithm is simpler to implement.   
 Can halt the loop where ever we want.   
 No memory constrains.   
   
3.1.3 Character Recognition   
Algorithms used :   
A. Contour Analysis   
B. Neural Network   
C. Correlation Formula   
   
A. Contour Analysis   
   
 The Contour Analysis allows to describe, store, compare and find the objects   
presented in the form of the exterior outlines - contours.   
 At first, we define such an object contour. The contour is a boundary of object, a   
population of points (pixels), separating object from a background.   
 In systems of computer vision, some formats of coding of a contour are used - the   
code of Freeman, two-dimensional coding, polygonal coding are most known. But   
all these formats of coding are not used in a CA.   
 Instead, in a CA the contour is encoded by the sequence consisting of complex   
numbers. On a contour, the point which is called as starting point is fixed. Then,   
the contour is scanned (is admissible - clockwise), and each vector of offset is   
noted by a complex number a+ib. Where a - point offset on x axis, and b - offset   
on y axis. Offset is noted concerning the previous point.   
   
   
   
   
   
   
   
   
 Owing to the physical nature of three-dimensional objects, their contours are   
always closed and cannot have self-intersection. The last vector of a contour   
always leads to the starting point.   
 Each vector of a contour we will name elementary vector (EV). And sequence of   
complex-valued numbers - vector-contour (VC).   
 Vectors-contours we will designate the big Greek letters, and their elementary a   
vector - small Greek letters.   
 Thus, vector-contour Γ of length k can be designated as:   
   
   
 As scalar product of contours, Γ and N are called such complex number:   
   
   
 Where k - dimensionality of a VC, γn - n the elementary vector of contour Γ, νn - n   
EV of contour N. (γn, νn) - the scalar product of complex numbers calculated as:   
   
 The scalar product of usual vectors and scalar product of complex numbers –   
 If we multiplied an EV as simple a vector, their scalar product would look so:   
   
Compare this formula to the formula (2) and you note that:   
   
Outcome of scalar product of vectors is the real number. And outcome of product   
of complex numbers - a complex number.   
   
The real part of scalar product of complex numbers coincides with scalar product   
of appropriate vectors. That is complex product includes vectorial scalar product.   
   
Let's introduce one more concept - the normalized scalar product (NSP):   
   
   
   
   
   
   
Where |Γ| and |N| - the norms (length) of contours calculated as:   
   
   
   
   
The NSP in space of complex numbers, also is a complex number.   
   
Thus, unity is greatest possible value of norm of NSP (it follows from a Cauchy–  
Bunyakovsky–Schwarz inequality: |ab| <= |a||b|), and it is reached only if...   
   
   
   
...where μ - the arbitrary complex number.   
   
Let's introduce the concept of intercorrelation function (ICF) of two contours:   
   
   
   
Where N(m) - a contour received from N by cycle shift by its EV on m of elements.   
   
For an example, if N = (n1, n2, n3, n4), N(1) = (n2, n3, n4, n1), N(2) = (n3, n4, n1,   
n2) and so on.   
   
Values of this function show contours Γ and N are how much similar if to shift   
starting point N on m positions.   
   
Let's discover the magnitude having the maximum norm among values an ICF:   
   
   
From determinations a NSP and an ICF, it is clear that τmax is a measure of   
similarity of two contours, invariant to transposition, scaling, rotation and starting   
point shift.   
   
Thus, the norm |τmax| shows a level of similarity of contours, and reaches unity for   
identical contours, and the argument arg(τmax) gives an angle of rotation of one   
contour, concerning another.   
   
Let's introduce one more concept - an autocorrelation function (ACF). The   
Autocorrelation function is an ICF for which N=Γ. As a matter of fact is a scalar   
product of a contour most on itself at various shifts of starting point:   
   
 For this purpose, we take the image a size n\*n pixels. Then breed its uniform grid   
with a step s. The total length of all grid lines is:   
   
 It turns out that passage from the plane two-dimensional image to contours does   
not reduce dimensionality of the task. We as before work in complexity O(n2).   
   
   
   
   
   
   
   
   
B. Neural Network   
The operations of the network implementation in this project can be summarized by   
the following steps:   
 Training phase   
o Analyze image for characters   
o Convert symbols to pixel matrices   
o Retrieve corresponding desired output character and convert to Unicode   
o Lineraize matrix and feed to network   
o Compute output   
o Compare output with desired output Unicode value and compute error   
o Adjust weights accordingly and repeat process until preset number of iterations   
   
 Testing phase   
o Analyze image for characters   
o Convert symbols to pixel matrices   
o Compute output   
o Display character representation of the Unicode output   
Essential components of the implementation are:   
   
Formation of the network and weight initialization routine   
   
Pixel analysis of images for symbol detection   
   
Loading routines for training input images and corresponding desired output   
characters in special files named character trainer sets (\*.cts)   
   
Loading and saving routines for trained network (weight values)   
   
Character to binary Unicode and vice versa conversion routines   
   
   
   
   
   
Error, output and weight calculation routines   
   
1. Network Formation   
   
   
   
   
   
   
   
   
   
   
The MLP Network implemented for the purpose of this project is composed of 3   
layers, one input, one hidden and one output.   
The input layer constitutes of 150 neurons which receive pixel binary data from a   
10x15 symbol pixel matrix. The size of this matrix was decided taking into   
consideration the average height and width of character image that can be mapped   
without introducing any significant pixel noise.   
The hidden layer constitutes of 250 neurons whose number is decided on the basis of   
optimal results on a trial and error basis.   
The output layer is composed of 16 neurons corresponding to the 16-bits of Unicode   
encoding.   
To initialize the weights a random function was used to assign an initial random   
number which lies between two preset integers named ±weight\_bias. The weight bias   
is selected from trial and error observation to correspond to average weights for quick   
convergence.   
2. Symbol image detection   
The process of image analysis to detect character symbols by examining pixels is the   
core part of input set preparation in both the training and testing phase. Symbolic   
extents are recognized out of an input image file based on the color value of individual   
pixels, which for the limits of this project is assumed to be either black RGB   
   
   
   
   
(255,0,0,0) or white RGB (255,255,255,255). The input images are assumed to be in   
bitmap form of any resolution which can be mapped to an internal bitmap object in the   
Microsoft Visual Studio environment. The procedure also assumes the input image is   
composed of only characters and any other type of bounding object like a boarder line   
is not taken into consideration.   
The procedure for analyzing images to detect characters is listed in the following   
algorithms:   
i. Determining character lines   
Enumeration of character lines in a character image („page‟) is essential in   
delimiting the bounds within which the detection can proceed. Thus detecting the   
next character in an image does not necessarily involve scanning the whole   
image all over again.   
Algorithm:   
1. start at the first x and first y pixel of the image pixel(0,0), Set number of lines   
to 0   
2. scan up to the width of the image on the same y-component of the image   
a. if a black pixel is detected register y as top of the first line   
b. if not continue to the next pixel   
c. if no black pixel found up to the width increment y and reset x to scan the   
next horizontal line   
3. start at the top of the line found and first x-component pixel(0,line\_top)   
4. scan up to the width of the image on the same y-component of the image   
a. if no black pixel is detected register y-1 as bottom of the first line.   
Increment number of lines   
b. if a black pixel is detected increment y and reset x to scan the next   
horizontal line   
5. start below the bottom of the last line found and repeat steps 1-4 to detect   
subsequent lines   
6. If bottom of image (image height) is reached stop.   
   
ii. Detecting Individual symbols   
Detection of individual symbols involves scanning character lines for   
orthogonally separable images composed of black pixels.   
Algorithm:   
1. start at the first character line top and first x-component   
2. scan up to image width on the same y-component   
a. if black pixel is detected register y as top of the first line   
b. if not continue to the next pixel   
   
   
   
   
3. start at the top of the character found and first x-component,   
pixel(0,character\_top)   
4. scan up to the line bottom on the same x-component   
a. if black pixel found register x as the left of the symbol   
b. if not continue to the next pixel   
c. if no black pixels are found increment x and reset y to scan the next   
vertical line   
5. start at the left of the symbol found and top of the current line,   
pixel(character\_left, line\_top)   
6. scan up to the width of the image on the same x-component   
a. if no black characters are found register x-1 as right of the symbol   
b. if a black pixel is found increment x and reset y to scan the next vertical   
line   
7. start at the bottom of the current line and left of the symbol,   
pixel(character\_left,line\_bottom)   
8. scan up to the right of the character on the same y-component   
a. if a black pixel is found register y as the bottom of the character   
b. if no black pixels are found decrement y and reset x to scan the next   
vertical line   
   
Fig 3. Line and Character boundary detection   
From the procedure followed and the above figure it is obvious that the detected   
character bound might not be the actual bound for the character in question. This   
is an issue that arises with the height and bottom alignment irregularity that   
exists with printed alphabetic symbols. Thus a line top does not necessarily mean   
top of all characters and a line bottom might not mean bottom of all characters as   
well.   
Hence a confirmation of top and bottom for the character is needed.   
An optional confirmation algorithm implemented in the project is:   
A. start at the top of the current line and left of the character   
B. scan up to the right of the character   
1. if a black pixels is detected register y as the confirmed top   
2. if not continue to the next pixel   
   
   
   
   
3. if no black pixels are found increment y and reset x to scan the next   
horizontal line   
   
Fig 4. Confirmation of Character boundaries   
   
3. Symbol Image Matrix Mapping   
The next step is to map the symbol image into a corresponding two dimensional binary   
matrix. An important issue to consider here will be deciding the size of the matrix. If   
all the pixels of the symbol are mapped into the matrix, one would definitely be able to   
acquire all the distinguishing pixel features of the symbol and minimize overlap with   
other symbols. However this strategy would imply maintaining and processing a very   
large matrix (up to 1500 elements for a 100x150 pixel image). Hence a reasonable   
tradeoff is needed in order to minimize processing time which will not significantly   
affect the separability of the patterns. The project employed a sampling strategy which   
would map the symbol image into a 10x15 binary matrix with only 150 elements.   
Since the height and width of individual images vary, an adaptive sampling algorithm   
was implemented. The algorithm is listed below:   
Algorithm:   
a. For the width (initially 20 elements wide)   
1. Map the first (0,y) and last (width,y) pixel components directly to the first (0,y)   
and last (20,y) elements of the matrix   
2. Map the middle pixel component (width/2,y) to the 10th matrix element   
3. subdivide further divisions and map accordingly to the matrix   
b. For the height (initially 30 elements high)   
1. Map the first x,(0) and last (x,height) pixel components directly to the first   
(x,0) and last (x,30) elements of the matrix   
2. Map the middle pixel component (x,height/2) to the 15th matrix element   
3. subdivide further divisions and map accordingly to the matrix   
c. Further reduce the matrix to 10x15 by sampling by a factor of 2 on both the width   
and the height   
   
   
   
   
   
Fig. 5 Mapping symbol images onto a binary matrix   
In order to be able to feed the matrix data to the network (which is of a single   
dimension) the matrix must first be linearized to a single dimension. This is   
accomplished with a simple routine with the following algorithm:   
1. start with the first matrix element (0,0)   
2. increment x keeping y constant up to the matrix width   
a. map each element to an element of a linear array (increment array index)   
b. if matrix width is reached reset x, increment y   
3. repeat up to the matrix height (x,y)=(width, height)   
Hence the linear array is our input vector for the MLP Network. In a training phase all   
such symbols from the trainer set image file are mapped into their own linear array and   
as a whole constitute an input space. The trainer set would also contain a file of   
character strings that directly correspond to the input symbol images to serve as the   
desired output of the training. A sample mini trainer set is shown below:   
   
   
Fig. 6 Input Image and Desired output text files for the sample Mini-Tahoma   
trainer set   
   
A. Training   
Once the network has been initialized and the training input space prepared the   
network is ready to be trained. Some issues that need to be addressed upon training the   
network are:   
   
How chaotic is the input space? A chaotic input varies randomly and in extreme   
range without any predictable flow among its members.   
   
   
   
   
   
How complex are the patterns for which we train the network? Complex patterns   
are usually characterized by feature overlap and high data size.   
   
What should be used for the values of:   
o Learning rate   
o Sigmoid slope   
o Weight bias   
   
How many Iterations (Epochs) are needed to train the network for a given number   
of input sets?   
   
What error threshold value must be used to compare against in order to   
prematurely stop iterations if the need arises?   
Alphabetic optical symbols are one of the most chaotic input sets in pattern   
recognitions studies. This is due to the unpredictable nature of their pictorial   
representation seen from the sequence of their order. For instance the Latin alphabetic   
consecutive character „A‟ and „B‟ have little similarity in feature when represented in   
their pictorial symbolic form. The figure below demonstrates the point of chaotic and   
non-chaotic sequence with the Latin and some factious character set:   
   
Fig. 7 Example of chaotic and non-chaotic symbol sequences   
The complexity of the individual pattern data is also another issue in character   
recognition. Each symbol has a large number of distinct features that need to be   
accounted for in order to correctly recognize it. Elimination of some features might   
result in pattern overlap and the minimum amount of data required makes it one of the   
most complex classes of input space in pattern recognition.   
Other than the known issues mentioned, the other numeric parameters of the network   
are determined in real time. They also vary greatly from one implementation to   
another according to the number of input symbols fed and the network topology.   
For the purpose of this project the parameters use are:   
   
Learning rate = 150   
   
Sigmoid Slope = 0.014   
   
Weight bias = 30 (determined by trial and error)   
   
Number of Epochs = 300-600 (depending on the complexity of the font types)   
   
Mean error threshold value = 0.0002 (determined by trial and error)   
   
Algorithm:   
The training routine implemented the following basic algorithm   
1. Form network according to the specified topology parameters   
   
   
   
   
2. Initialize weights with random values within the specified ±weight\_bias value   
3. load trainer set files (both input image and desired output text)   
4. analyze input image and map all detected symbols into linear arrays   
5. read desired output text from file and convert each character to a binary Unicode   
value to store separately   
6. for each character :   
a. calculate the output of the feed forward network   
b. compare with the desired output corresponding to the symbol and compute   
error   
c. back propagate error across each link to adjust the weights   
7. move to the next character and repeat step 6 until all characters are visited   
8. compute the average error of all characters   
9. repeat steps 6 and 8 until the specified number of epochs   
a. Is error threshold reached? If so abort iteration   
b. If not continue iteration   
Flowchart:   
The flowchart representation of the algorithm is illustrated below   
   
   
   
   
   
   
B. Testing   
The testing phase of the implementation is simple and straightforward. Since the   
program is coded into modular parts the same routines that were used to load, analyze   
and compute network parameters of input vectors in the training phase can be reused   
in the testing phase as well.   
The basic steps in testing input images for characters can be summarized as follows:   
Algorithm:   
   
load image file   
   
analyze image for character lines   
   
for each character line detect consecutive character symbols   
o analyze and process symbol image to map into an input vector   
o feed input vector to network and compute output   
o convert the Unicode binary output to the corresponding character and render to   
a text box   
   
   
   
Flowchart:   
   
   
   
   
   
   
C. Character Recognition Using Correlation   
   
Correlation Coefficient Definition:-   
   
A measure of the strength of linear association between two variables.   
 Correlation will always be in between -1.0 and +1.0.If the correlation is   
positive ,we have a positive relationship.If the correlation is negative ,we have   
a negative relationship.   
   
Formula:-   
   
Correlation(r) =[ NΣXY - (ΣX)(ΣY) / Sqrt([NΣX2 - (ΣX)2][NΣY2 - (ΣY)2])]   
   
   
   
   
   
   
   
Where   
   
   
   
N=Number of pixel in image.   
   
   
   
X=Pixel in First image.   
   
   
   
Y=Pixel in Second image.   
   
   
   
∑XY=Sum of product of First and Second Pixels.   
   
   
   
∑X=Sum of First Image Pixels.   
   
   
   
∑Y=Sum of Second Image Pixels.   
   
   
   
∑X2=Sum of square of Pixel in First Image.   
   
   
   
∑Y2=Sum of square of Pixels in Second Image.   
   
3.2 Cropping an Image   
 Cropping refers to the removal of the outer parts of an image to improve framing, or   
change aspect ratio.   
 The   
program   
takes   
an   
image   
and   
the   
cropping   
parameters   
as   
input.   
Then it determines if the cropping area lies within the image or not. In case a cropping   
area in portion or fully lies outside the main image co-ordinates, the program adjusts   
the crop area.   
   
The following picture shows the cropping are lying within the image.   
   
   
   
   
   
 Image shows portions of cropping area lying outside the original image.   
 It takes an image and the cropping parameters as input.   
Then it determines if the cropping area lies within the image or not. In case a cropping   
area in portion or fully lies outside the main image co-ordinates, it adjusts the crop   
area.   
 The parameters used in the cropping are:   
– Height of the cropping rectangle.   
– Width of the cropping rectangle.   
– X Co-ordinate of the start point of the cropping rectangle.   
– Y Co-ordinate of the start point of the cropping rectangle.   
 It also takes care of the negative co-ordinates supplied for the crop area rectangle.   
 Finally it crops the input image and saves a copy of the cropped image.   
   
   
   
   
   
CHAPTER-4 SYSTEM SPECIFICATION   
4.1 Specification   
4.1.1 Input Specification   
1. Scanned document image file format supported : bmp, tif/tiff, pgm and jpeg. -   
Input Image : Graylevel, black 'n' white or colored.   
   
2. Image dimensions : Upto (3500 × 3500) pixels.   
 supported. - Minimum scanning resolution : 300dpi.   
   
3. Maximum scanning resolution : 600dpi.   
   
4. Input image can contain text/graphics/picture. - Maximum input skew is   
 15degrees.   
   
5. Input image scanned in portrait/landscape mode.   
   
   
4.1.2 Output Specification   
1. OCR processed output   
2. Database Engine is used to store the processed output to database   
3. Presentation Engine presents the data from database in appropriate format to make   
sure that the data has been successfully saved.   
   
4.1.3 Functional Specifications   
The proposed OCR system specifications, as per the common conclusive decision by   
the members of the project group are listed below :   
• Pre-processing:   
 Detecting skew and corrected: A maximum skew angle of 15 degrees is   
supported.   
 Binarization: Adaptive thresholding based techniques for good binarization   
results.   
 Text Direction Recognition : Text scanned in both portrait and landscape   
mode will be supported.   
 Image Cropping utility will be provided.   
 Automatic determination of scanning resolution (desirable).   
 Pre-processing color images (desirable).   
   
   
   
   
   
   
   
• Page Segmentation and Layout Analysis:   
 Classification and segmenting page into text/non-text regions of gray level   
images.   
 Segmentation of color pages (desirable).   
 Determining the page layout and semantic labeling (desirable).   
 Non Text region classification as picture and graphics.   
 Multi page documentation (desirable).   
 Line and Word segmentation in script independent fashion.   
• Word Segmentation:   
 Script-independent line and word boundary detection. - Script-dependent word   
boundary detection scheme.   
• Symbol Recognition:   
 Script based component identification.   
 Touching and broken symbol processing.   
 Provision for reject class for unknown symbols. - Provision for classifier   
combination.   
• Text Recognition:   
 Unicode generation for recognized symbols.   
 Use of script/language models for ambiguities or error resolution in   
classification (desirable).   
 Dictionary based error correction.   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
4.2 Actor Definition   
The user of the system gives input document image to the OCR, using a scanner or selects an   
input image from the database. The OCR system processes the input image and displays the final   
output on the presentation engine. The system administrator has control over the Image   
Acquisition interface, OCR system, and the presentation engine.   
   
   
   
Junior Clerk  
Input Image  
Image Processing  
\*  
\*  
Save  
Edit  
PreProcessing  
Segmentation  
Character  
Reconization  
Template And Table  
defination  
Includes  
Includes  
Includes  
Head Clerk  
\*  
\*  
\*  
\*  
\*  
\*  
\*  
\*  
«uses»  
\*  
\*  
\*  
\*  
System  
   
   
   
   
   
   
   
   
4.2.1 Junior Clerk   
Description:- He can convert the documents of his interest into electronically accessible   
format. Annotate the document image for future use. He can create a database of document   
images   
through   
scanning   
He can save the image in BMP, TIFF, PGM, JPG formats. He can explore the various   
features of the OCR system like manual segmentation, language selection, etc.   
Aliases:- System User, Customer, Client   
Inherits:- None   
Actor Type:- Active   
   
   
4.2.2 Head Clerk   
Description:- He enters the details for the new Template to be formed. He enters the   
fields which needs to be stored in the database.   
Aliases:- System User, Customer, Client   
Inherits:- None   
Actor Type:- Active   
   
4.2.3 Image Input   
Description:- This use case is used to get the image from the junior clerk. The image   
which is scanned by the clerk and stored in the database. This image then Undergoes all   
Image Processing steps.   
Aliases:- None   
Inherits:- None   
   
Actor Type: Passive   
   
4.2.4 Image processing   
Description:- This use case is used to Process the image the clerk has entered. This Use   
Case also contains many steps to be performed. The Steps are: Pre-processing   
Segmentation, Character Recognition.   
Aliases:- None   
Inherits:- None   
Actor Type:- Passive   
   
   
   
   
   
   
4.2.5 Save   
Description:- This use case is used to save the information so obtained from the Image   
Processing Step. The data obtained from Image Processing step is displayed to the user   
and on click of SAVE the user is able to save the obtained data to Database.   
Aliases:- None   
Inherits:- None   
Actor Type:- Passive   
   
4.2.6 Edit   
Description:- This use case is used to Edit the data so obtained from Image processing   
step. The data is displayed to the user, in case of any changes that the user manually   
wants to make, he has to click on Edit to manually make the changes.   
Aliases:- None   
Inherits:- None   
Actor Type:- Passive   
   
   
4.2.7 Pre-Processing   
Description:- This use case is part of the Image Processing. This is the one of the step   
followed during Image Processing. In pre-processing there are number of steps that are   
under gone they are: Converting Image to Grey Scale, Noise Removal, Skew Removal.   
Aliases:- None   
Inherits:- None   
Actor Type:- Passive   
   
4.2.8 Segmentation   
Description:- This use case is part of the Image Processing. This is the one of the step   
followed during Image Processing. In Segmentation the image is cut into segments to   
identify different shapes and lines.   
Aliases:- None   
Inherits:- None   
Actor Type:- Passive   
   
   
   
   
   
   
   
4.2.9 Character Recognition   
Description:- This use case is part of the Image Processing. This is the one of the step   
followed during Image Processing. In Character Recognition the character are recognised   
and are given labels so using these labels the character so obtained is stored into database.   
Aliases:- None   
Inherits:- None   
Actor Type:- Passive   
   
4.2.10 Table And Template Defination   
Description:- This use case is used to generate new Templates for new format. The Head   
Clerk enters the sample image and enters the necessary fields whose values needed to be   
obtained during Image Processing.   
Aliases:- None   
Inherits:- None   
Actor Type:- Passive   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
4.3 Tables   
   
 TABLE 4.3.1: MASTER\_TEMPLATE\_TABLE   
FIELD NAME   
DATA TYPE   
CONSTRAINTS   
DESCRIPTION   
Sample\_ Image   
Image   
Not Null   
Image of the printed   
documents   
showing   
the template design.   
Template\_Name   
Text   
Not Null   
Name given to the   
template.   
Template\_ID   
VarChar   
Primary Key, Auto   
Unique ID given to   
each template.   
Printable\_Length   
Numeric   
Not Null   
Length of the image   
showing   
only   
printable area.   
Printable\_Breadth   
Numeric   
Not Null   
Breadth of the image   
showing   
only   
printable area.   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
 TABLE 4.3.2: CHILD\_TEMPLATE\_TABLE   
FIELD NAME   
DATA TYPE   
CONSTRAINTS   
DESCRIPTION   
Template\_ID   
VarChar   
Foreign Key   
   
Field\_Name   
Text   
Not Null   
Name of the field   
according to which   
data   
is   
to   
be   
segregated.   
Field\_ID   
VarChar   
Primary Key, Auto   
Unique ID given to   
each field.   
Field\_Length   
Numeric   
Not Null   
Length of the field   
where that particular   
data   
will   
be   
available.   
Field\_Breadth   
Numeric   
Not Null   
Breadth of the field   
where that particular   
data   
will   
be   
available.   
Field\_Type   
Text   
Not Null   
What type of data the   
field will be having.   
Number\_Of\_Characters Numeric   
Not null   
Specifies   
the   
Maximum Characters   
appearing   
in   
the   
field.   
Left\_Spacing   
Numeric   
Not Null   
Shows how far the   
field is from Left   
Boundary   
of   
printable area.   
Top\_Spacing   
Numeric   
Not Null   
Shows how far the   
field is from Top   
Boundary   
of   
printable area.   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
TABLE 4.3.3: VALUE\_TABLE   
FIELD NAME   
DATA TYPE   
CONSTRAINTS   
DESCRIPTION   
Image   
Image   
Not Null   
Stores the image of   
the   
printed   
documents   
from   
where data has to be   
accessed and stored   
into database.   
Template\_name   
Text   
Not Null   
Gives the name of the   
template to be used.   
Template\_ID   
VarChar   
Auto   
Unique ID of the   
template to be used.   
Field\_Name   
Text   
Not Null   
Name of the field   
which is used.   
Field\_ID   
VarChar   
Auto   
Unique ID of the   
Field to be used.   
Value   
VarChar   
Not Null   
Value regarding that   
field.   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
CHAPTER-5 RESULT   
According to our project, we have performed various steps for storing printed documents into   
database.   
Various steps performed are shown as follow:   
1. First we have converted RGB image into GreyScale ,we have used inbuilt function   
colorConvertOp().   
   
   
2. Then after we have binarize the image ,we have used threshold value concept in which we   
have set threshold value below threshold value image will become black i.e. 0 value and   
above threshold image will become white i.e. 255.   
   
   
   
   
   
   
3. Then after we crop the required field from the image ,as explained in cropping part above   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
CHAPTER-6 CONCLUSION   
6.1 Conclusion   
Our Project is on Transforming printed documents to Database. So it input of scan printed   
document is given and that image of scan documents is converted to text form according to   
user requirement. Thus our project looked upon the problems faced by Educational Institutes,   
for storing data which is in Printed format. It helped in reducing manual burden and is also   
less time consuming. This software can also be use for any kind of format of printed   
documents as our software provides dynamic definition of templates. And according to the   
templates added by the admin user the data is processed by the software and gives the   
appropriate answer. Thus not only Educational Institutes are benefited by this but also the   
government offices or any other organizations using such printed documents are benefited.   
We looked upon various algorithms and techniques for pre processing and character   
recognition from a image and implemented most optimal ones amongst them,thus resulting in   
more speed and accuracy.   
This makes our project dynamic and is feasible for any kind of organization.   
We have successfully completed our project.   
   
6.2 Scope Of Future Enhancement   
This project can be further extended for recognizing handwritten documents.This software can   
be further upgraded in which functionality can be added to train handwriting of a particular   
individual and then can be used to recognize documents written by that individual.Also   
software can be trained to recognize handwriting of multiple individuals and also different   
fonts.There is also scope of increasing accuracy of the recognizer so that no manual watch   
should be needed on the software other than inputting the data.Thus software can be   
automized to a higher level.   
   
   
   
   
   
   
   
   
   
   
   
   
CHAPTER-7 REFRENCES   
   
BOOKS:-   
1. DIGITAL IMAGE PROCESSING BY A.GONZALES   
WEBSITE:-   
1. http://www.ieeexplore.com   
2. http://www.codeproject.com   
3. http://www.fadooengineers.com   
4. http://www.scribd.com   
5. Using Neural Networks to Create an Adaptive Character Recognition System   
© 2002, Alexander J. Faaborg   
Cornell University, Ithaca NY.   
6. http://www.stackoverflow.com