# Optical Character Recognition in MATLAB

By

**Saurav Tomar**
mc/2k12/63

## Abstract

In this ambitions project I plan to implement an optical character recognition engine in matlab.
To sum it up , the user inputs an image into matlab, the matlab code them does some image processing on the input image, reduces the noise and then disects the image into possible character objects, and then for each object found in the image, the code tries to guess the most probable character match from a pre-processed set of images.
Output is a word stored in a text file.

## Problem Description

Before OCR can be used, the source material must be scanned using an optical scanner (and sometimes a specialized circuit board in the PC) to read in the image as a bitmap (a pattern of dots).

Software to recognize the images is also required.

The character recognition software then processes these scans to differentiate between images and text and determine what letters are represented in the light and dark areas. Older OCR systems match these images against stored bitmaps based on specific fonts. The hit -or-miss results of such pattern-recognition systems helped establish OCR's reputation for inaccuracy.

The image scanned is then converted to a matrix in matlab, matrix is a separate datatype to contain image data in matlab to process them further.

After converting the image to a matrix, noise reduction algorithms are run so as to reduce the noise in the image, and rmove all redundant objects from he image.

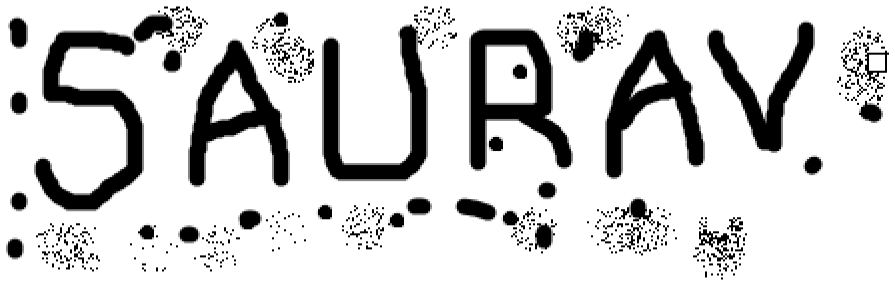
After removing the noise the image is then disected into objects, and each object is treated as a separate image for the further processes.

For each object found in the image, the sub-image is then compared to template images of all the character and alphabets, and the template image which resembles the sub-image to the greatest extent is  identified as the most probable match.

## Solution Approach

### Read Image

```
I = imread('training.bmp');
imshow(I)
```



Input image with noise

### Basic Image processing to remove noise

```
% Convert to gray scale
if size(imagen,3)==3 %RGB image
   imagen=rgb2gray(imagen);
end

% Convert to BW
threshold = graythresh(imagen);
imagen =~im2bw(imagen,threshold);

% Remove all object containing fewer than 30 pixels
imagen = bwareaopen(imagen,30);

% Show image
imshow(imagen);
```

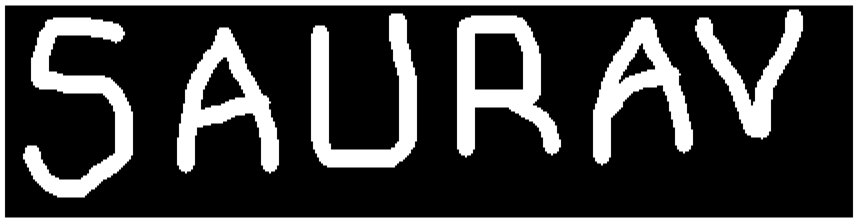


Image after noise removal

## Disecting image into character objects and detecting most probable match from templates

```
imgn = imagen;
[L Ne] = bwlabel(imgn);
for n=1:Ne
    [r,c] = find(L==n);
    % Extract letter
    n1=imgn(min(r):max(r),min(c):max(c));
    % Resize letter (same size of template)
    img_r=imresize(n1,[42 24]);
    %Uncomment line below to see letters one by one
    imshow(img_r);pause(0.5)
    %------------------------------------------------------------------
    % Call fcn to convert image to text
    letter=read_letter(img_r,num_letras);
    % Letter concatenation
    word=[word letter];
end
```

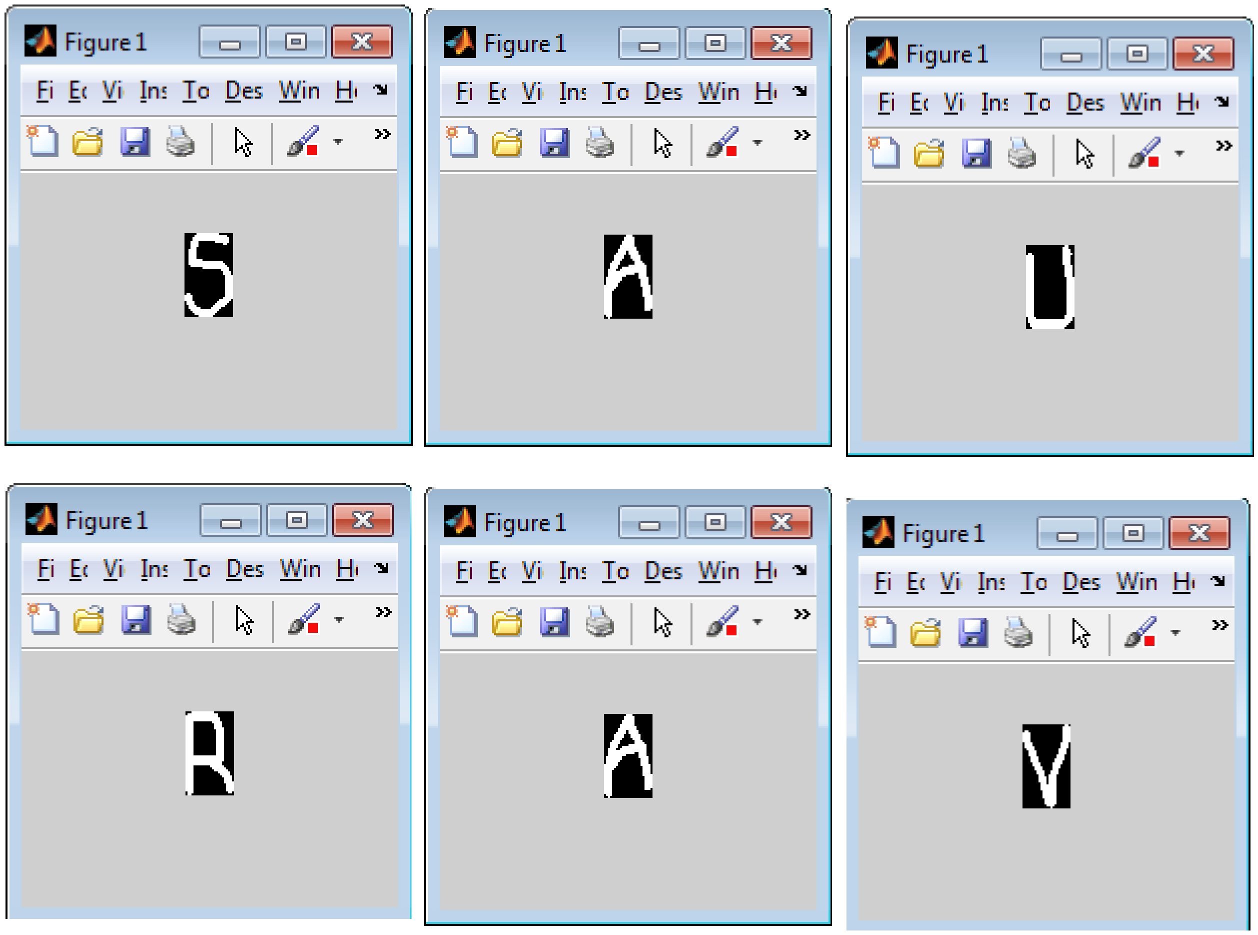


*Image dissected into objects*

## Creating character templates

```
%CREATE TEMPLATES Letter

A=imread('letters_numbers\A.bmp');B=imread('letters_numbers\B.bmp');
C=imread('letters_numbers\C.bmp');D=imread('letters_numbers\D.bmp');
E=imread('letters_numbers\E.bmp');F=imread('letters_numbers\F.bmp');
G=imread('letters_numbers\G.bmp');H=imread('letters_numbers\H.bmp');
I=imread('letters_numbers\I.bmp');J=imread('letters_numbers\J.bmp');
K=imread('letters_numbers\K.bmp');L=imread('letters_numbers\L.bmp');
M=imread('letters_numbers\M.bmp');N=imread('letters_numbers\N.bmp');
O=imread('letters_numbers\O.bmp');P=imread('letters_numbers\P.bmp');
Q=imread('letters_numbers\Q.bmp');R=imread('letters_numbers\R.bmp');
S=imread('letters_numbers\S.bmp');T=imread('letters_numbers\T.bmp');
U=imread('letters_numbers\U.bmp');V=imread('letters_numbers\V.bmp');
W=imread('letters_numbers\W.bmp');X=imread('letters_numbers\X.bmp');
Y=imread('letters_numbers\Y.bmp');Z=imread('letters_numbers\Z.bmp');

%Number
one=imread('letters_numbers\1.bmp');  two=imread('letters_numbers\2.bmp');
three=imread('letters_numbers\3.bmp');four=imread('letters_numbers\4.bmp');
five=imread('letters_numbers\5.bmp'); six=imread('letters_numbers\6.bmp');
seven=imread('letters_numbers\7.bmp');eight=imread('letters_numbers\8.bmp');
nine=imread('letters_numbers\9.bmp'); zero=imread('letters_numbers\0.bmp');

%*-*-*-*-*-*-*-*-*-*-*-
letter=[A B C D E F G H I J K L M...
   N O P Q R S T U V W X Y Z];

number=[one two three four five...
   six seven eight nine zero];

character=[letter number];

templates=mat2cell(character,42,[24 24 24 24 24 24 24 ...
   24 24 24 24 24 24 24 ...
   24 24 24 24 24 24 24 ...
   24 24 24 24 24 24 24 ...
   24 24 24 24 24 24 24 24]);

save ('templates','templates')

clear all
```

## Comparing user input with character templates to find most probable match

```
function letter=read_letter(imagn,num_letras)

% Computes the correlation between template and input image
% and its output is a string containing the letter.
% Size of 'imagn' must be 42 x 24 pixels
% Example:
% imagn=imread('D.bmp');
% letter=read_letter(imagn)

global templates
comp=[ ];
for n=1:num_letras
   sem=corr2(templates{1,n},imagn);
   comp=[comp sem];
end

vd=find(comp==max(comp));
%*-*-*-*-*-*-*-*-*-*-*-*-*-*-
if vd==1
   letter='A';
elseif vd==2
   letter='B';

. . .
```

## Saving the word to a text file

```
%save word to result.txt
fprintf(fid,'%s\n',word);
fclose(fid);

%Open 'result.txt' file
winopen('result.txt')
clear all
```

## Conslusion

This OCR program shows good results for character images, even with images having considerable noise.Although , the program shows varying results depending on various characters, this problem can be solved by increasing the images in character template database.

It was a overwhelming experience to work on this project, having ported this project from javascript to matlab, I got to learn a lot about the amazing Image Processing Toolbox of matlab, and its immense possibilities.

*Saurav Tomar*
*1st May 2014*