Shri Vile Parle Kelavani Mandal's
**DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING**
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

***Department of Computer Engineering***
***Academic Year 2023-2024***

Name: Akshat Parekh SAP ID: 60004200116
Name: Dhruv Gandhi SAP ID: 60004200133
Name: Devang Shah SAP ID: 60004200158
Class: BE Computer Engineering Division: C - 21

*Digital Signal Processing Mini Project*

# Illumination-Resilient OCR

## AIM:

To design an Optical Character Recognition that can read text from an image. Also, reduce the noise to achieve conversion of any form of text or text-containing images such as printed or scanned text images, into an editable text format.

## THEORY:

Text extraction from images is a critical task with wide-ranging applications in our digital age. Before OCR technology was available, the only option to digitize printed paper documents was by manually re-typing the text. Not only was this massively time-consuming, but it also came with inaccuracy and typing errors.

This project focuses on employing Digital Signal Processing (DSP) techniques to tackle the challenge of converting text in images into human-readable text format. This capability is crucial for applications such as Optical Character Recognition (OCR), document digitization, and data retrieval from images. In an era dominated by visual content, the ability to unlock textual information from images has far-reaching implications, from simplifying data entry to making historical manuscripts accessible in a digital format.

The hurdles are manifold, including the diversity of fonts, text sizes, backgrounds, and image qualities. Variability in factors like lighting, resolution, image orientation, and noise levels adds complexity to this task.

The crucial step involves converting the image to grayscale, ensuring that further processing is done in a single channel. Image thresholding follows, where Otsu's method is used to determine an optimal threshold, and the image is binarized, creating a binary image with text as the foreground.

The code then performs morphological processing to remove small objects, potentially noise, from the binary image. It uses the 'bwareaopen' function for this purpose. After preprocessing, the code initiates text extraction. It defines a 'word' variable to accumulate the extracted text and opens a text file named 'text.txt' for writing. The code loads predefined templates for character recognition. A loop is entered for text extraction. In each iteration, lines of text are separated from the binary image, and connected components are labeled and counted. For each component, the corresponding letter is extracted, resized to match predefined templates, and recognized using a function named 'read_letter.' The recognized letter is added to the 'word' variable. After processing all connected components in a line, 'word' is written to the 'text.txt' file. This loop continues until there's no more text in 're,' and 'word' is cleared for the next line. Finally, the 'text.txt' file is obtained, allowing the extracted text to be accessed.

## CODE:

create_templates.m

```
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
```

lines.m

```
function [fl re]=lines(im_texto)
% Divide text in lines
% im_texto->input image; fl->first line; re->remain line
im_texto=clip(im_texto);
num_filas=size(im_texto,1);
for s=1:num_filas
if sum(im_texto(s,:))==0
nm=im_texto(1:s-1, :); % First line matrix
rm=im_texto(s:end, :);% Remain line matrix
fl = clip(nm);
re=clip(rm);
break
else
fl=im_texto;
re=[ ];
end
end
function img_out=clip(img_in)
[f c]=find(img_in);
img_out=img_in(min(f):max(f),min(c):max(c));
```

read_letter.m

```
function letter=read_letter(imagn,num_letras)
% Computes the correlation between template and input image
% and its output is a string containing the letter.
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';
elseif vd==3
    letter='C';
elseif vd==4
    letter='D';
elseif vd==5
    letter='E';
elseif vd==6
    letter='F';
elseif vd==7
    letter='G';
elseif vd==8
    letter='H';
elseif vd==9
    letter='I';
elseif vd==10
    letter='J';
```

```
elseif vd==11
    letter='K';
elseif vd==12
    letter='L';
elseif vd==13
    letter='M';
elseif vd==14
    letter='N';
elseif vd==15
    letter='O';
elseif vd==16
    letter='P';
elseif vd==17
    letter='Q';
elseif vd==18
    letter='R';
elseif vd==19
    letter='S';
elseif vd==20
    letter='T';
elseif vd==21
    letter='U';
elseif vd==22
    letter='V';
elseif vd==23
    letter='W';
elseif vd==24
    letter='X';
elseif vd==25
    letter='Y';
elseif vd==26
    letter='Z';

elseif vd==27
    letter='1';
elseif vd==28
    letter='2';
elseif vd==29
    letter='3';
elseif vd==30
    letter='4';
elseif vd==31
    letter='5';
elseif vd==32
    letter='6';
elseif vd==33
    letter='7';
elseif vd==34
    letter='8';
elseif vd==35
    letter='9';
else
    letter='0';
end
```

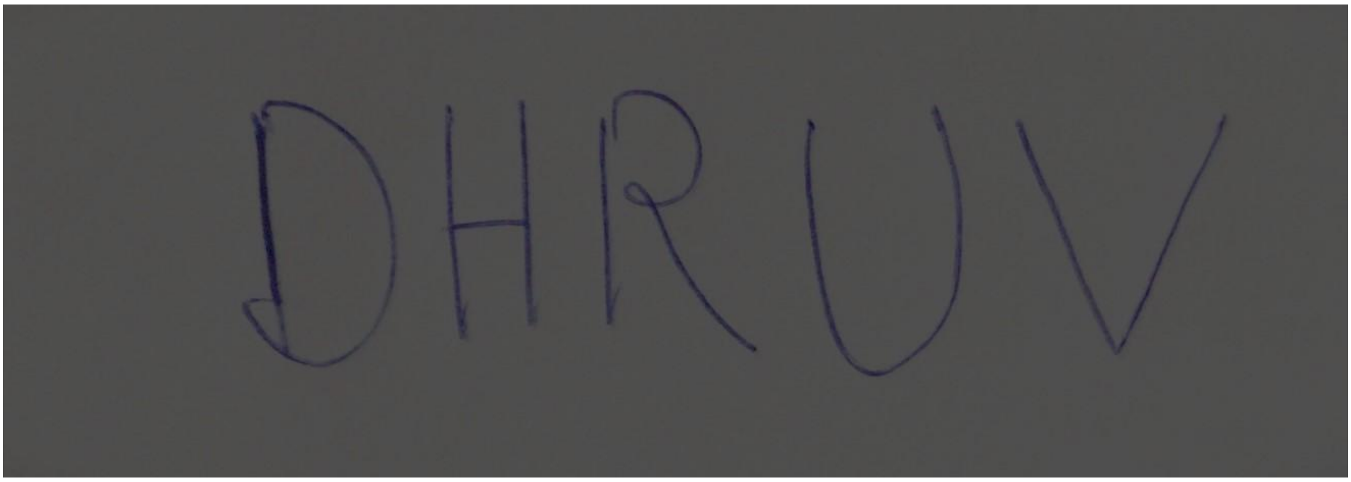
## OCR.m

```
% Read image
imagen=imread('TEST_1.JPG');
% Show image
figure(1)
imshow(imagen);
title('INPUT IMAGE WITH NOISE')

% Convert to gray scale
if size(imagen,3)==3 %RGB image
imagen=rgb2gray(imagen);
end
% Show image
figure(2)
imshow(imagen);
title('INPUT IMAGE GRAYSCALED')

% use median filter
%imagen = medfilt2(imagen,[15 15]);
%figure(3);
%imshow(imagen);
%title('AFTER APPLYING MEDIAN FILTER')

% Convert to BW
threshold = graythresh(imagen);
imagen =~im2bw(imagen,threshold);
title('AFTER THRESHOLDING')
% Show image
figure(4);
imshow(imagen);

% Remove all object containing fewer than 30 pixels
% Opening - Morphological processing
imagen = bwareaopen(imagen,30);
% Show image
figure(5);
imshow(imagen);
title('AFTER MORPHOLOGICAL PROCESSING')

%Storage matrix word from image
word=[ ];
re=imagen;

%Opens text.txt as file for write
fid = fopen('text.txt', 'wt');
% Load templates
load templates
global templates

% Compute the number of letters in template file
num_letras=size(templates,2);
while 1
    %Fcn 'lines' separate lines in text
    [fl re]=lines(re);
    imgn=fl;
    % Label and count connected components
    [L Ne] = bwlabel(imgn);
    for n=1:Ne
        [r,c] = find(L==n);
```

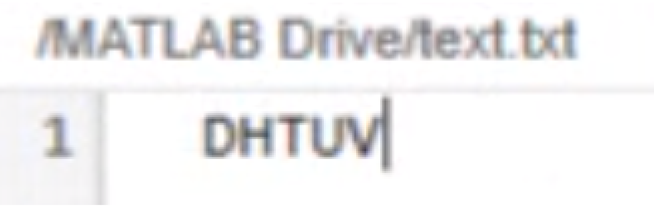
```
  % Extract letter
    n1=imgn(min(r):max(r),min(c):max(c));
  % Resize letter (same size of template)
    img_r=imresize(n1,[42 24]);
  % Call fcn to convert image to text
    letter=read_letter(img_r,num_letras);
  % Letter concatenation
    word=[word letter];
  end
  %Write 'word' in text file
  fprintf(fid,'%s\n',word);
  % Clear 'word' variable
  word=[ ];
  %*When the sentences finish, breaks the loop
  if isempty(re)
    break
  end
  end

fclose(fid);
%Open 'text.txt' file
winopen('text.txt')
clear all
```

## OUTPUT:

Input Image: (Low Contrast, Low Vignette, Low Exposure)

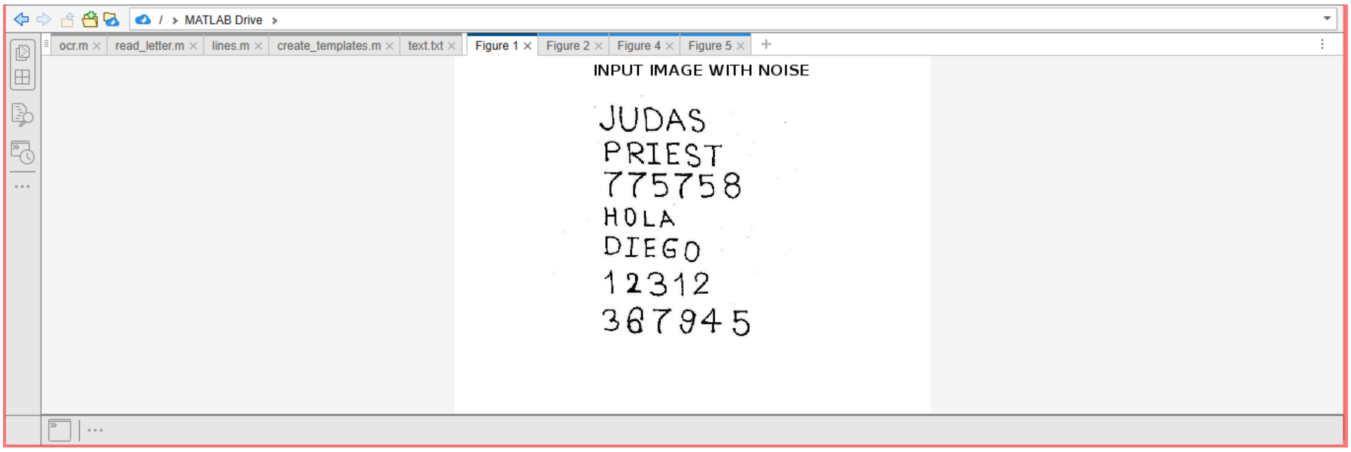


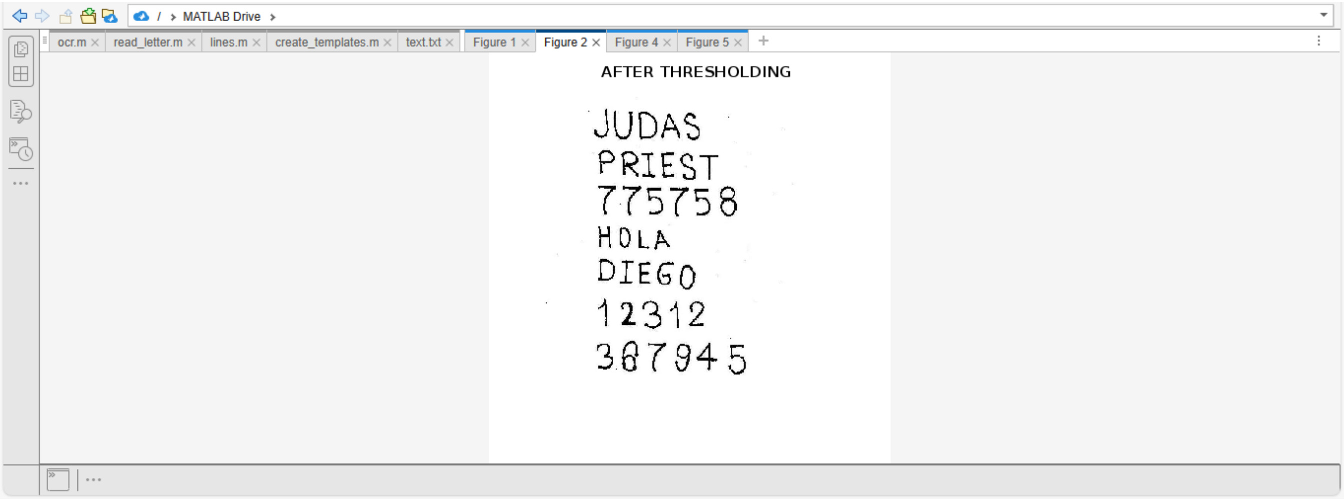
After Low Contrast, Low Vignette, Low Exposure
Output read by ocr :

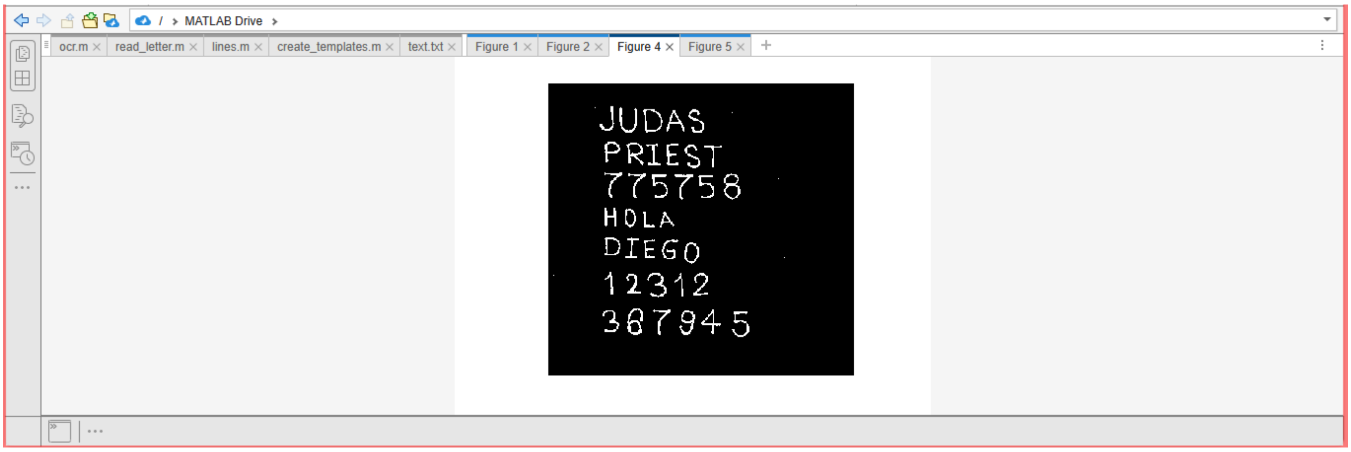


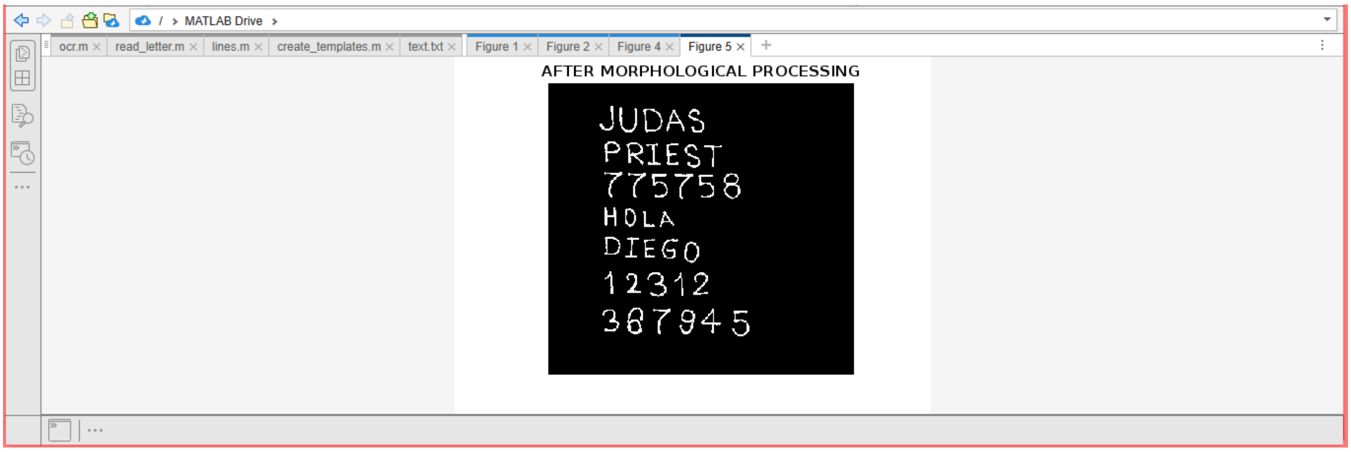
```
/MATLAB Drive/text.txt
1     DHTUV
```

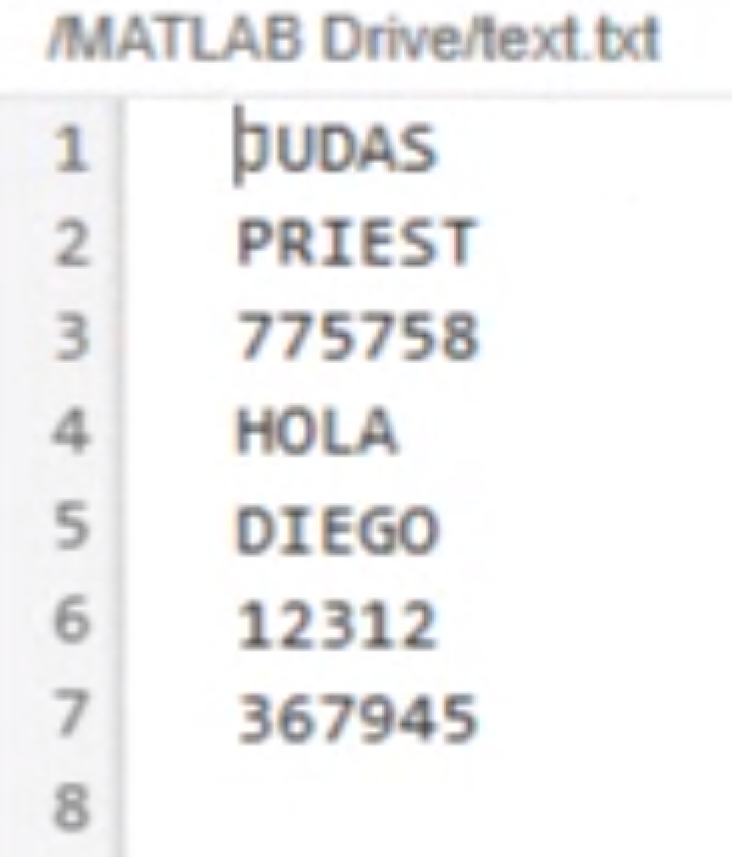
2. Low Contast Input Image :











/MATLAB Drive/text.txt

```
JUDAS
PRIEST
775758
HOLA
DIEGO
12312
367945

```

**CONCLUSION:** This project harnesses Digital Signal Processing (DSP) techniques to address the significant challenge of text extraction from images. With wide-ranging applications in Optical Character Recognition (OCR), document digitization, and data retrieval, this technology provides a crucial bridge between the visual and textual worlds. Despite the complexities posed by varying fonts, sizes, backgrounds, and image quality, the project successfully converts images to binary representations, cleans them using morphological operations, and extracts text from connected components. The output, stored in 'text.txt,' enables easy access to the extracted text.