**Practical No: - 8**

**AIM: -Case study- Optical Character Reorganisation is using MATLAB.**

**What is OCR?****

Next to keypunching, Optical Character Recognition is the oldest data entry technique in existence. Long before the first key-to-disk system of CRT was used; Optical Character Readers were entering data in commercial and government EDP installations.

The popularity of OCR has been increasing each year with the advent of fast microprocessors providing the vehicle for vastly improved recognition techniques. This can be shown in OCR wands now reading print that, over 10 years ago, large batch readers would have rejected. There have also been tremendous improvements in increasing both effective read rates and accuracy. Data Entry through OCR is faster, more accurate, and generally more efficient than keystroke data entry. Desktop OCR scanners can read typewritten data into a computer at rates up to 2400 words per minute!

**How Does OCR Work?**

There are two basic methods used for OCR: Matrix matching and feature extraction. Of the two ways to recognize characters, matrix matching is the simpler and more common.

Matrix Matching compares what the OCR scanner sees as a character with a library of character matrices or templates. When an image matches one of these prescribed matrices of dots within a given level of similarity, the computer labels that image as the corresponding ASCII character.

Feature Extraction is OCR without strict matching to prescribed templates. Also known as Intelligent Character Recognition (ICR), or Topological Feature Analysis, this method varies by how much "computer intelligence" is applied by the manufacturer. The computer looks for general features such as open areas, closed shapes, diagonal lines, line intersections, etc. This method is much more versatile than matrix matching. Matrix matching works best when the OCR encounters a limited repertoire of type styles, with little or no variation within each style. Where the characters are less predictable, feature, or topographical analysis is superior.

**OCR Fonts**

What is a font? A font is the term given to a set of characters, usually 0 - 9, A through Z, and a few special characters. Each character within a font will have a defined reproducible size and shape. For OCR, these are defined by ANSI, the American National Standards Institute.

OCR fonts, or characters, that can be read by the lower speed, lower cost systems we are discussing here require well defined character shapes that are very reproducible and designed to be both machine and human readable. These unique and well defined character sets allow for greater accuracy.

[**OCR Scanners**](http://www.dataid.com/productsocr.htm)

OCR reading devices are fundamentally classified with two categories, Text Input and Data Capture.

Text input devices are page readers or document scanners that scan entire documents or large portions of documents. The source data is entered with the intention of someone editing it during or after it is scanned. Text input devices have varying degrees of automation from hand fed to having automatic feeding, reading, sorting, and stacking capabilities.

Data Capture devices are designed to capture repetitive data and to perform formatting functions on the data as it is being entered. The data delivered from the scanner to the computer must be very accurate because it is entered without the intention of being edited later, so accuracy must be higher than text input.

**Program:**

warning off

clc, close all, clear all

imagen=imread('TEST\_1.jpg');

imshow(imagen);

title('INPUT IMAGE WITH NOISE')

if size(imagen,3)==3

imagen=rgb2gray(imagen);

end

threshold = graythresh(imagen);

imagen =~im2bw(imagen,threshold);

imagen = bwareaopen(imagen,30);

word=[ ];

re=imagen;

fid = fopen('text.txt', 'wt');

load templates

global templates

num\_letras=size(templates,2);

while 1

[fl re]=lines(re);

imgn=fl;

[L Ne] = bwlabel(imgn);

for n=1:Ne

[r,c] = find(L==n);

n1=imgn(min(r):max(r),min(c):max(c));

img\_r=imresize(n1,[42 24]);

letter=read\_letter(img\_r,num\_letras);

word=[word letter];

end

fprintf(fid,'%s\n',word);

word=[ ];

if isempty(re)

break

end

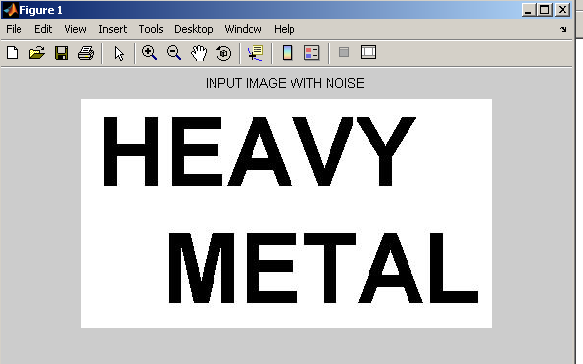
end

fclose(fid);

winopen('text.txt')

clear all

**Input File:**



**Output: -**

