Why text extraction is so important?

1. It has many useful applications such as automatic bank check processing
2. vehicle license plate recognition
3. document analysis and page segmentation
4. signboard detection and translation
5. content based image indexing, assistance to visually impaired persons,
6. text translation system for foreigners

The image content is classified into two categories:

1. **Perceptual Contents** :
   Perceptual contents include colors, shapes, textures, intensities, and their temporal changes while semantic contents include objects, events, and their relations.
2. **Semantic Contents** :
   Text content contains high level of semantic information as compared to visual information.

Text appearing in images is classified into three categories:

1. **Document Text**
   A document image usually contains text and few graphic components. It is acquired by Scanning journal, printed document, handwritten historical document, and book cover etc.
2. **Caption Text**
   It is also known as overlay text or artificial text. It is artificially superimposed on the image at the time of editing, like subtitles and it usually describes the subject of the image content.
3. **Scene Text**
   It occurs naturally as a part of the scene image and contain important semantic information such as advertisements, names of streets, institutes, shops, road signs, traffic information, board signs, nameplates, food containers, street signs, bill boards, banners, and text on vehicle

Properties of Text in Images

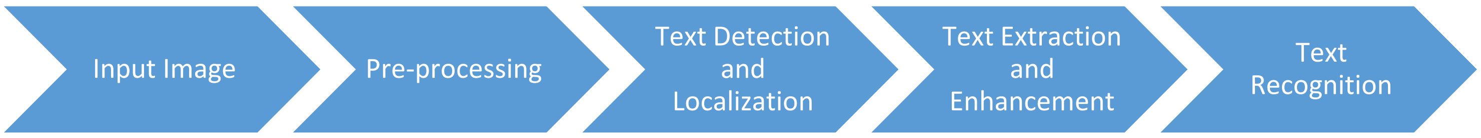
Text in images exhibit variations due to the difference in the following properties

- **Size**: The size of text may vary a lot.
- **Alignment**: Scene text may be aligned in any direction and have geometric distortions while caption text usually aligned horizontally and sometimes may appear as non-planar text.
- **Color**: The characters tend to have same or similar color but low contrast between text and background makes text extraction difficult.
- **Edge**: Most caption and scene texts are designed to be easily read, hence resulting in strong edges at the boundaries of text and background.
- **Compression**: Many images are recorded, transferred, and processed in compressed format. Thus, a faster text extraction system can be achieved if one can extract text without decompression.

- **Distortion**: Due to changes in camera angles, some text may carry perspective distortions that affect extraction performance.

Process of Text Extraction:

The input image may be gray scale or color, compressed on uncompressed format. Text detection refers to the determination of the presence of text in the image while text localization is the process of determining the location of text and generating bounding boxes around it. After that, text is extracted i.e. segmented from the background. Enhancement of the extracted text is required as the text region usually has low-resolution and is prone to noise. Thereafter, the extracted text can be recognized using OCR.


Input Image
Pre-processing
Text Detection and Localization
Text Extraction and Enhancement
Text Recognition

TEXT EXTRACTION TECHNIQUES

PERFORMANCE ANALYSIS FOR A TECHNIQUES

Various parameters are used to analyze the performance of text extraction techniques and given as follow:

$$1.\ Detection\ Rate(DR) = \frac{correct\ detected\ text}{Ground\ truth\ text}$$

$$2.\ Precision\ Rate(PR) = \frac{correct\ detected}{Correct\ Detected + false\ positives}$$

$$3.\ Recall\ Rate(RR) = \frac{correct\ detected}{Correct\ detected + false\ negatives}$$

$$4.\ False\ Alarm\ Rate(FAR) = \frac{no.of\ text\ blocks\ falsely\ detected}{Total\ no.of\ text\ blocks}$$

The various techniques of text extraction are as follow:

1. **Region based Method:**
   Region-based method uses the properties of the color or gray scale in the text region or their differences to the corresponding properties of the background. They are based on the fact that there is very little variation of color within text and this color is sufficiently distinct from text's immediate background. Text can be obtained by thresholding the image at intensity level in between the text color and that of its immediate background. This method is not robust to complex background. This method is further divided into two sub -approaches: connected component (CC) and edge based.
   - **CC based Method:**
   CC-based methods use a bottom-up approach by grouping small components into successively larger components until all regions are identified in the image. A geometrical analysis is required

to merge the text components using the spatial arrangement of those components so as to filter out non-text components and the boundaries of the text regions are marked. This method locate locates text quickly but fails for complex background.

- **Edge based Method:**

Edges are a reliable feature of text regardless of color/intensity, layout, orientations, etc. Edge based method is focused on high contrast between the text and the background. The three distinguishing characteristics of text embedded in images that can be used for detecting text are edge strength, density and the orientation variance. Edge based text extraction algorithm is a general-purpose method, which can quickly and effectively localize and extract the text from both document and indoor/ outdoor images. This method is not robust for handling large size text.

2. **Texture based Method:**

   This method uses the fact that text in images have discrete textural properties that distinguish them from the background. The techniques based on Gabor filters, Wavelet, Fast Fourier transform (FFT), spatial variance, etc. Are used to detect the textual properties of the text region in the image. This method is able to detect the text in the complex background. The only drawback of this method is large computational complexity in texture classification stage.

3. **Morphological based Method:**

   Mathematical morphology is a topological and geometrical based method for image analysis. Morphological feature extraction techniques have been efficiently applied to character recognition and document analysis. It is used to extract important text contrast features from the processed images. These features are invariant against various geometrical image changes like translation, rotation, and scaling. Even after the lightning condition or text color is changed, the feature still can be maintained. This method works robustly under different image alterations.

CODE

```
i=imread(image location');
imshow(i);
title('INPUT IMAGE WITH NOISE')
if size(i,3)==3
    i=rgb2gray(i);
end
threshold = graythresh(i);
i =~im2bw(i,threshold);
i = bwareaopen(i,30);
word=[ ];
re=i;
fid = fopen('text.txt', 'wt');
load templates
global templates
num_letters=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')
```

RECOGNITION

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

LINE SEGMENTATION

```
function [fl re]=lines(im_texto)
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, :);
        rm=im_texto(s:end, :);
        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));
```

TEMPLATES

```
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');
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')
```