Roll Number : 21BCE092, 21BCE105, 21BCE156, 21BCE157

Course Code and Name : 2CS501 and Machine Learning

Innovative Assignment

[ ]:
```
import cv2                                          #importing libraries
import numpy as np
import matplotlib.pyplot as plt
import sys
import time
```

[ ]:
```
data =cv2.CascadeClassifier('haarcascade_russian_plate_number.xml')
```

[ ]:
```
def plt_show(image, title="", gray = False, size =(100,100)):
    temp = image
    if gray == False:
        temp = cv2.cvtColor(temp, cv2.COLOR_BGR2RGB)
        plt.title(title)
        plt.imshow(temp, cmap='gray')
        plt.show()
```

Convert image to Gray Scale

[ ]:
```
def detect_number(img):
    temp = img
    gray = cv2.cvtColor(temp,cv2.COLOR_BGR2GRAY)
    number = data.detectMultiScale(gray,1.2)
    print("number plate detected: "+str(len(number)))
    for numbers in number:
        (x,y,w,h) = numbers
        roi_gray = gray[y:y+h, x:x+w]
        roi_color = img[y:y+h, x:x+h]
        cv2.rectangle(temp, (x,y), (x+w,y+h), (0,255,0), 3)
    plt_show(temp)
```

Taking input Image of Car

[ ]:
```
img = cv2.imread("car.jpg")            #Input Image of Car
plt_show(img)
# detect_number(img)
```

[ ]:
```
import cv2 as cv
from google.colab.patches import cv2_imshow
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
cv2_imshow(gray)
```

Morphological Transform

[20]:
```
import numpy as np
kernel = np.ones((5,5), np.uint8)
erosion = cv2.erode(img, kernel, iterations = 1)
plt.subplot(1,1,1), plt.imshow(erosion)
plt.title('Morphological Transformation'), plt.xticks([]), plt.yticks([])
plt.show()
```

[ ]:
```
import imutils
image = img
ratio = image.shape[0] / 500.0
orig = image.copy()
image = imutils.resize(image, height = 500)
```

[ ]:
```
gray = cv2.cvtColor(image, cv2.COLOR_RGB2GRAY)      #Applying gray scale
gray = cv2.GaussianBlur(gray, (5, 5), 0)            #Gaussian Blur
edged = cv2.Canny(gray, 75, 200)                    #Canny Edge Detection
cv2_imshow(image)
```

[ ]:
```
cnts = cv2.findContours(edged.copy(), cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
  cnts = imutils.grab_contours(cnts)
  cnts = sorted(cnts, key = cv2.contourArea, reverse = True)[:10]
  screenCnt = None
```

[ ]:
```
for c in cnts:
        # approximate the contour
        peri = cv2.arcLength(c, True)
        approx = cv2.approxPolyDP(c, 0.018 * peri, True)

        # if our approximated contour has four points, then
        # we can assume that we have found our screen
        print( len(approx) )
        if len(approx) == 4:
            screenCnt = approx
            break
```

```
6
6
8
13
11
```

```
11
9
7
7
8
```

[ ]:
```python
if screenCnt is None:
        detected = 0
        print("No contour detected")
else:
        detected = 1

if detected == 1:
        cv2.drawContours(img, [screenCnt], -1, (0, 255, 0), 3)
```

```
No contour detected
```

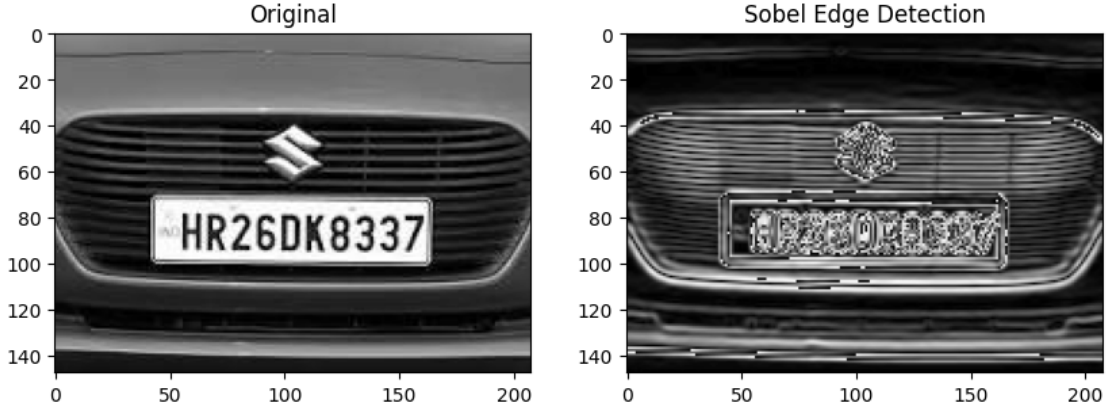
Edge Detection

[8]:
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
```

Sobel Edge Detection

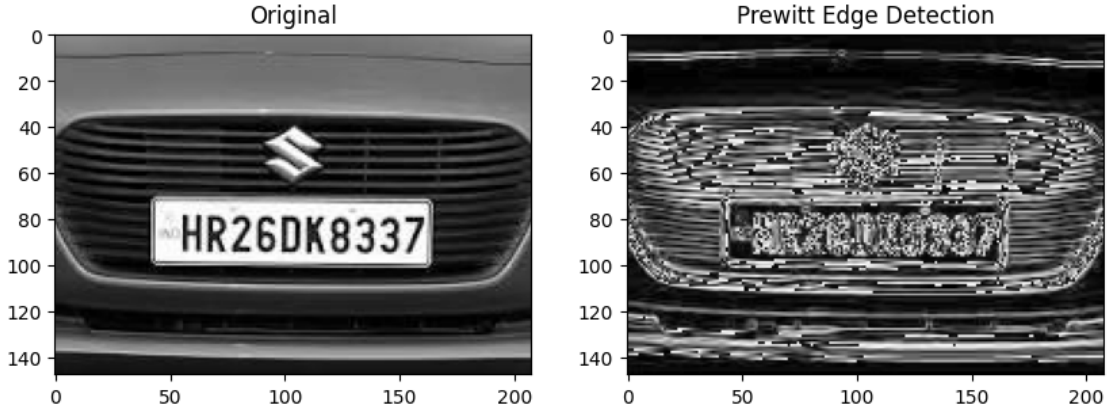
[9]:
```
image = 'car.jpg'
image = cv2.imread(image, cv2.IMREAD_GRAYSCALE)
blurred = cv2.GaussianBlur(image, (5, 5), 0)
sobelx = cv2.Sobel(blurred, cv2.CV_64F, 1, 0, ksize=3)
sobely = cv2.Sobel(blurred, cv2.CV_64F, 0, 1, ksize=3)
magnitude = np.sqrt(sobelx**2 + sobely**2)
magnitude = np.uint8(magnitude)
plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.imshow(image, cmap='gray')
plt.title('Original')
plt.subplot(1, 2, 2)
plt.imshow(magnitude, cmap='gray')
plt.title('Sobel Edge Detection')

plt.show()
```



Prewitt Edge Detection

[10]:
```
image_path = 'car.jpg'
image = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)
prewittx = cv2.Sobel(image, cv2.CV_64F, 1, 0, ksize=3)
prewitty = cv2.Sobel(image, cv2.CV_64F, 0, 1, ksize=3)
prewitt_edges = np.sqrt(prewittx**2 + prewitty**2)
prewitt_edges = np.uint8(prewitt_edges)
plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.imshow(image, cmap='gray')
plt.title('Original')
plt.subplot(1, 2, 2)
plt.imshow(prewitt_edges, cmap='gray')
plt.title('Prewitt Edge Detection')
plt.show()
```



Output and Accuracy

```
[ ]: !sudo apt install tesseract-ocr
```

```
Reading package lists… Done
Building dependency tree… Done
Reading state information… Done
The following additional packages will be installed:
  tesseract-ocr-eng tesseract-ocr-osd
The following NEW packages will be installed:
  tesseract-ocr tesseract-ocr-eng tesseract-ocr-osd
0 upgraded, 3 newly installed, 0 to remove and 10 not upgraded.
Need to get 4,816 kB of archives.
After this operation, 15.6 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy/universe amd64 tesseract-ocr-eng
all 1:4.00~git30-7274cfa-1.1 [1,591 kB]
Get:2 http://archive.ubuntu.com/ubuntu jammy/universe amd64 tesseract-ocr-osd
all 1:4.00~git30-7274cfa-1.1 [2,990 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy/universe amd64 tesseract-ocr amd64
4.1.1-2.1build1 [236 kB]
Fetched 4,816 kB in 0s (26.2 MB/s)
debconf: unable to initialize frontend: Dialog
debconf: (No usable dialog-like program is installed, so the dialog based
frontend cannot be used. at /usr/share/perl5/Debconf/FrontEnd/Dialog.pm line 78,
<> line 3.)
debconf: falling back to frontend: Readline
debconf: unable to initialize frontend: Readline
debconf: (This frontend requires a controlling tty.)
debconf: falling back to frontend: Teletype
dpkg-preconfigure: unable to re-open stdin:
Selecting previously unselected package tesseract-ocr-eng.
(Reading database … 120880 files and directories currently installed.)
Preparing to unpack …/tesseract-ocr-eng_1%3a4.00~git30-7274cfa-1.1_all.deb …
Unpacking tesseract-ocr-eng (1:4.00~git30-7274cfa-1.1) …
Selecting previously unselected package tesseract-ocr-osd.
Preparing to unpack …/tesseract-ocr-osd_1%3a4.00~git30-7274cfa-1.1_all.deb …
Unpacking tesseract-ocr-osd (1:4.00~git30-7274cfa-1.1) …
Selecting previously unselected package tesseract-ocr.
```

```
Preparing to unpack …/tesseract-ocr_4.1.1-2.1build1_amd64.deb …
Unpacking tesseract-ocr (4.1.1-2.1build1) …
Setting up tesseract-ocr-eng (1:4.00~git30-7274cfa-1.1) …
Setting up tesseract-ocr-osd (1:4.00~git30-7274cfa-1.1) …
Setting up tesseract-ocr (4.1.1-2.1build1) …
Processing triggers for man-db (2.10.2-1) …
```

[ ]:
```
!pip install pytesseract

import pytesseract
import shutil
import os
import random

try:
  from PIL import Image
except ImportError:
  import Image
from google.colab.patches import cv2_imshow
import cv2
from matplotlib import pyplot as plt
```

```
Collecting pytesseract
  Downloading pytesseract-0.3.10-py3-none-any.whl (14 kB)
Requirement already satisfied: packaging>=21.3 in
/usr/local/lib/python3.10/dist-packages (from pytesseract) (23.2)
Requirement already satisfied: Pillow>=8.0.0 in /usr/local/lib/python3.10/dist-
packages (from pytesseract) (9.4.0)
Installing collected packages: pytesseract
Successfully installed pytesseract-0.3.10
```

[ ]:
```
from google.colab import files
uploaded = files.upload()
```

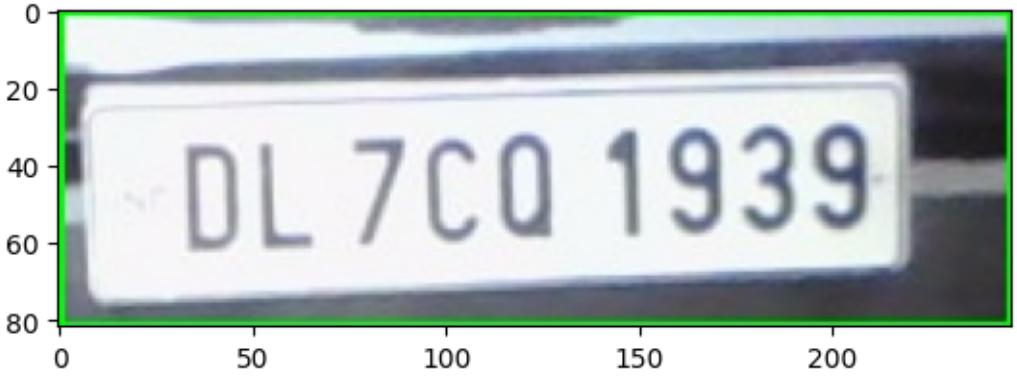
```
<IPython.core.display.HTML object>

Saving Image_0.jpg to Image_0.jpg
Saving Image_2.jpg to Image_2.jpg
Saving Image_4.jpg to Image_4.jpg
Saving Image_5.jpg to Image_5.jpg
Saving Image_6.jpg to Image_6.jpg
Saving Image_7.jpg to Image_7.jpg
Saving Image_8.jpg to Image_8.jpg
Saving Image_15.jpg to Image_15.jpg
```

[ ]:
```
img = cv2.imread('Image_0.jpg')
img1 = Image.open('Image_0.jpg')
cv2_imshow(img)
```

```
plt.imshow(img1)
ocrinfo = pytesseract.image_to_string(img)
print(ocrinfo)
```


```
DL7CcQ 1339)
```



[ ]:
```
img = cv2.imread('Image_2.jpg')
img1 = Image.open('Image_2.jpg')
cv2_imshow(img)
plt.imshow(img1)
ocrinfo = pytesseract.image_to_string(img)
print(ocrinfo)
```


```
TS 068 FM 66.)
```

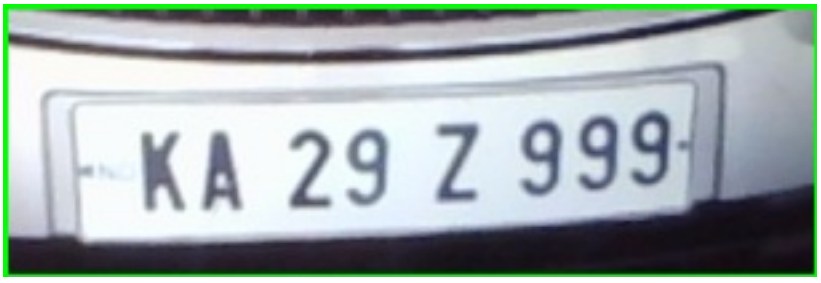
[ ]:
```
img = cv2.imread('Image_4.jpg')
img1 = Image.open('Image_4.jpg')
cv2_imshow(img)
plt.imshow(img1)
ocrinfo = pytesseract.image_to_string(img)
print(ocrinfo)
```


```
MH200V2366
--------
```

[ ]:
```
img = cv2.imread('Image_5.jpg')
img1 = Image.open('Image_5.jpg')
cv2_imshow(img)
plt.imshow(img1)
ocrinfo = pytesseract.image_to_string(img)
print(ocrinfo)
```


```
MH 20 EE 7602
```

[ ]:
```
img = cv2.imread('Image_7.jpg')
img1 = Image.open('Image_7.jpg')
cv2_imshow(img)
plt.imshow(img1)
ocrinfo = pytesseract.image_to_string(img)
print(ocrinfo)
```



```
-$$-$_____-_-

KA 29 Z 999)
```

[ ]:
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
```

[ ]:
```
l1 = [ [ 'DL7CQ1939', 'DL7CcQ133'] ,[ 'TS08FM8864', 'TS08FM64**'],[␣
↪'MH200V2366', 'MH200V2366'],[ 'MH20EE7602', 'MH20EE7602'],[ 'KA29Z999',␣
↪'KA29Z999']]

print(l1)
```

```
[['DL7CQ1939', 'DL7CcQ133'], ['TS08FM8864', 'TS08FM64**'], ['MH200V2366',
'MH200V2366'], ['MH20EE7602', 'MH20EE7602'], ['KA29Z999', 'KA29Z999']]
```

[ ]:
```
import numpy as np
arr = np.array(l1);

total = 0
right = 0
n = arr.shape[0];
m = arr.shape[1];

for i in range(0,n):

    total = total +  len(arr[i][0]);
    for j in range(0,len(arr[i][0])):
```

```
        if(arr[i][0][j] == arr[i][1][j]):
            right= right+1;

print(total)
print(right)

print("Accuracy of given Algorithm is: ",(right/total),"%")
```

```
47
39
Accuracy of given Algorithm is:  0.8297872340425532 %
```