PCV LAB – 4

ID:- 190030059

Name:- A.R.Snehita

PRELAB

1. Explain the operations done in edge detection in detail?

A. Edges are significant local changes of intensity in a digital image. An edge can be defined as a set of connected pixels that forms a boundary between two disjoint regions Edge Detection Operators are of two types: Gradient – based operator which computes first-order derivations in a digital image like, Sobel operator, Prewitt operator, Robert operator Gaussian – based operator which computes second-order derivations in a digital image like, Canny edge detector, Laplacian of Gaussian Sobel Operator: It is a discrete differentiation operator. It computes the gradient approximation of image intensity function for image edge detection. At the pixels of an image, the Sobel operator produces either the normal to a vector or the corresponding gradient vector. It uses two 3 x 3 kernels or masks which are convolved with the input image to calculate the vertical and horizontal derivative approximations respectively

2. Explain Region of Interest (ROI)?

A. A region of interest (ROI) is a subset of an image or a dataset identified for a particular purpose. The dataset could be any of the following: Waveform or 1D dataset: The ROI is a time or frequency interval on the waveform (a graph of some quantity plotted against time). Image or 2D dataset: The ROI is defined by given boundaries on an image of an object or on a drawing. Volume or 3D dataset: The ROI is the contours or the surfaces defining a physical object. Time-Volume or 4D dataset: Concerning the changing 3D dataset of an object changing in shape with time, the ROI is the 3D dataset during a specific time or period of time.

3. How many types of edges in image and why should we detect?

A. Edges are significant local changes of intensity in a digital image. An edge can be defined as a set of connected pixels that forms a boundary between two disjoint regions. There are three types of edges: Horizontal edges Vertical edges Diagonal edges Edge Detection is a method of segmenting an image into regions of discontinuity. It is a widely used technique in digital image processing like pattern recognition image morphology feature extraction Edge detection allows users to observe the features of an image for a significant change in the gray level. This texture indicating the end of one region in the image and the beginning of another. It reduces the amount of data in an image and preserves the structural properties of an image.

4. When should we use ROI?

A. Regarding the region of interest (ROI) as two- or threedimensional range is common in the computer-controlled image processing and also in imaging processes. In extreme cases, for a time span there occurs even a four-dimensional view. Often the region of interest is used in medical applications, especially in nuclear medicine. The procedure is also used in computer tomography. In this case the ROIs are multidimensional.

<u>INLAB:-</u>

1. Kushal is taking an extra class for edge detection there he got a work from his teacher to do all the task below. Kushal alone can’t do so help him to complete his work

i. Load the image.

ii. Convert the image into grayscale.

iii. Apply the gaussian blur to the image.

iv. Now apply the edge detection algorithm for the image.

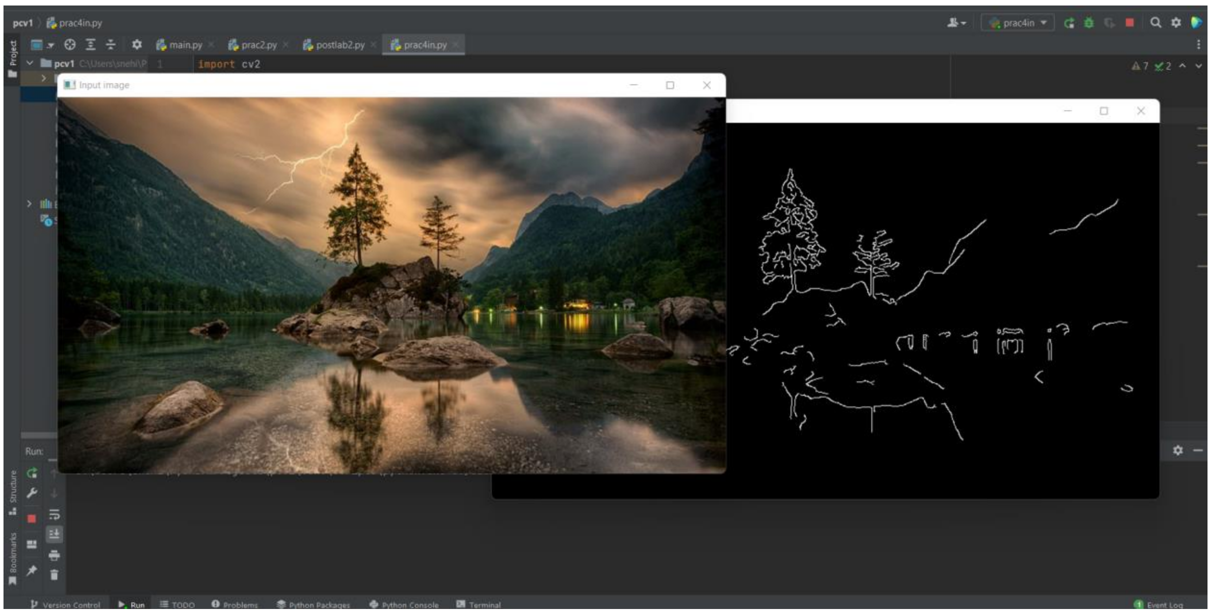
v. Then display the detected image.

CODE:-

```
import cv2
# Read an input image
image = cv2.imread(r'C:\Users\snehi\PycharmProjects\pcv1\pcvimage.jpg')
cv2.imshow("Input image",image)
gray = cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
Gaussian = cv2.GaussianBlur(gray,(7,7),0)
```

```
edges = cv2.Canny(image=Gaussian, threshold1=100, threshold2=200)
cv2.imshow("Canny edge Detection",edges)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

Screenshots:



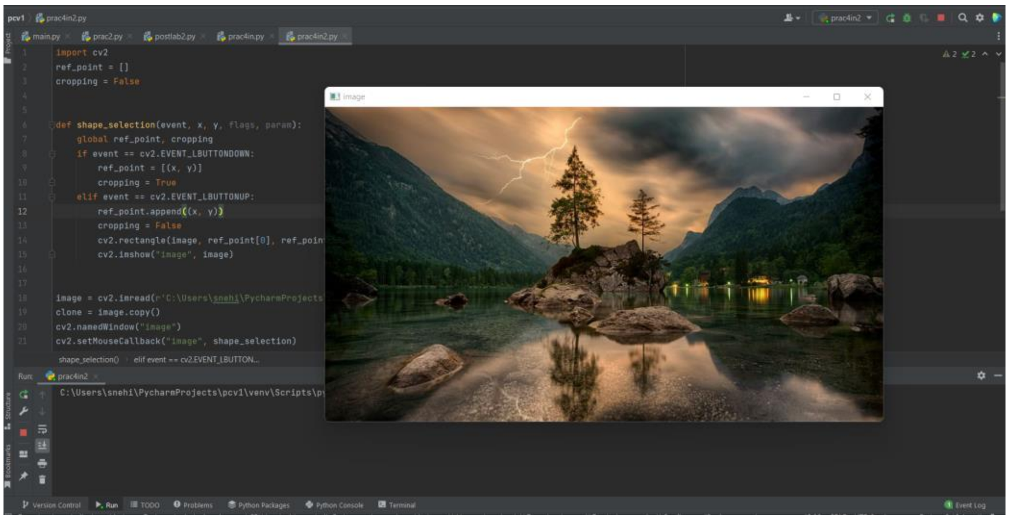
```
import cv2
```

2)Rurik and Kamal are friends. Rurik wants to crop an image and in which the region of image should selected by kamal and then rurik would like to show that selected region separated from original to kamal

```
import cv2
ref_point = []
cropping = False


def shape_selection(event, x, y, flags, param):
    global ref_point, cropping
    if event == cv2.EVENT_LBUTTONDOWN:
        ref_point = [(x, y)]
        cropping = True
    elif event == cv2.EVENT_LBUTTONUP:
        ref_point.append((x, y))
        cropping = False
        cv2.rectangle(image, ref_point[0], ref_point[1], (0, 255, 0), 2)
        cv2.imshow("image", image)


image = cv2.imread(r'C:\Users\snehi\PycharmProjects\pcv1\pcvimage.jpg')
clone = image.copy()
cv2.namedWindow("image")
```

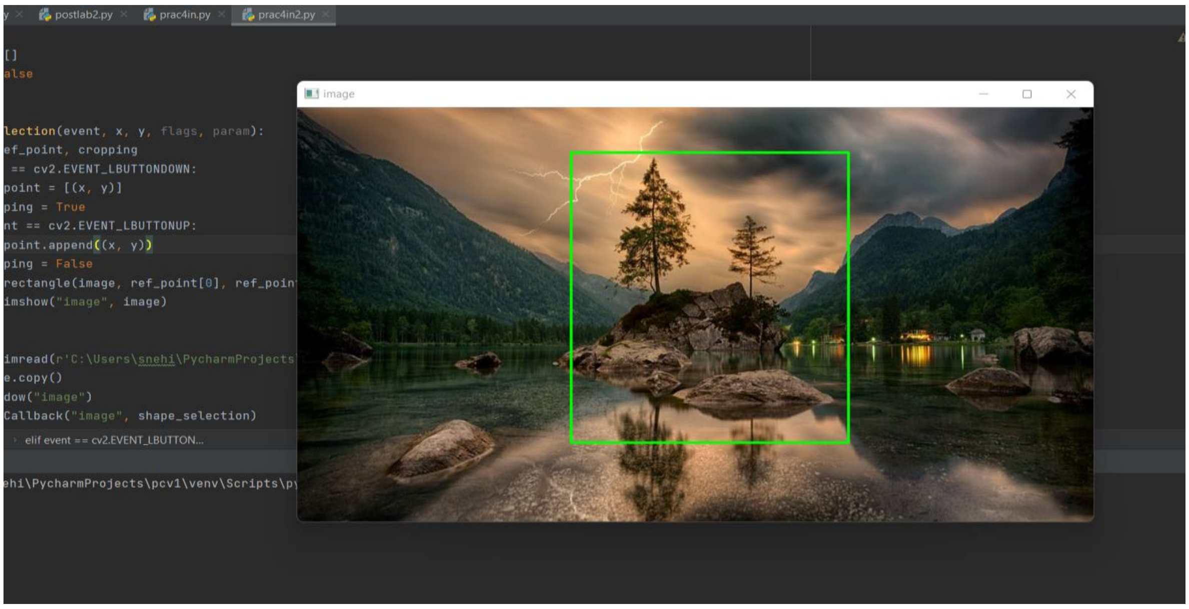
```python
cv2.setMouseCallback("image", shape_selection)
# keep looping until the 'q' key is pressed
while True:
    cv2.imshow("image", image)
    key = cv2.waitKey(1) & 0xFF
    if key == ord("r"):
        image = clone.copy()
    elif key == ord("c"):
        break

if len(ref_point) == 2:
    crop_img = clone[ref_point[0][1]:ref_point[1][1],
ref_point[0][0]:ref_point[1][0]]
    cv2.imshow("crop_img", crop_img)
    cv2.waitKey(0)
    cv2.destroyAllWindows()
```

Screenshots:-



```python
import cv2
ref_point = []
cropping = False


def shape_selection(event, x, y, flags, param):
    global ref_point, cropping
    if event == cv2.EVENT_LBUTTONDOWN:
        ref_point = [(x, y)]
        cropping = True
    elif event == cv2.EVENT_LBUTTONUP:
        ref_point.append((x, y))
        cropping = False
        cv2.rectangle(image, ref_point[0], ref_poin
        cv2.imshow("image", image)


image = cv2.imread(r'C:\Users\snehi\PycharmProjects
clone = image.copy()
cv2.namedWindow("image")
cv2.setMouseCallback("image", shape_selection)
```

```
C:\Users\snehi\PycharmProjects\pcv1\venv\Scripts\py
```



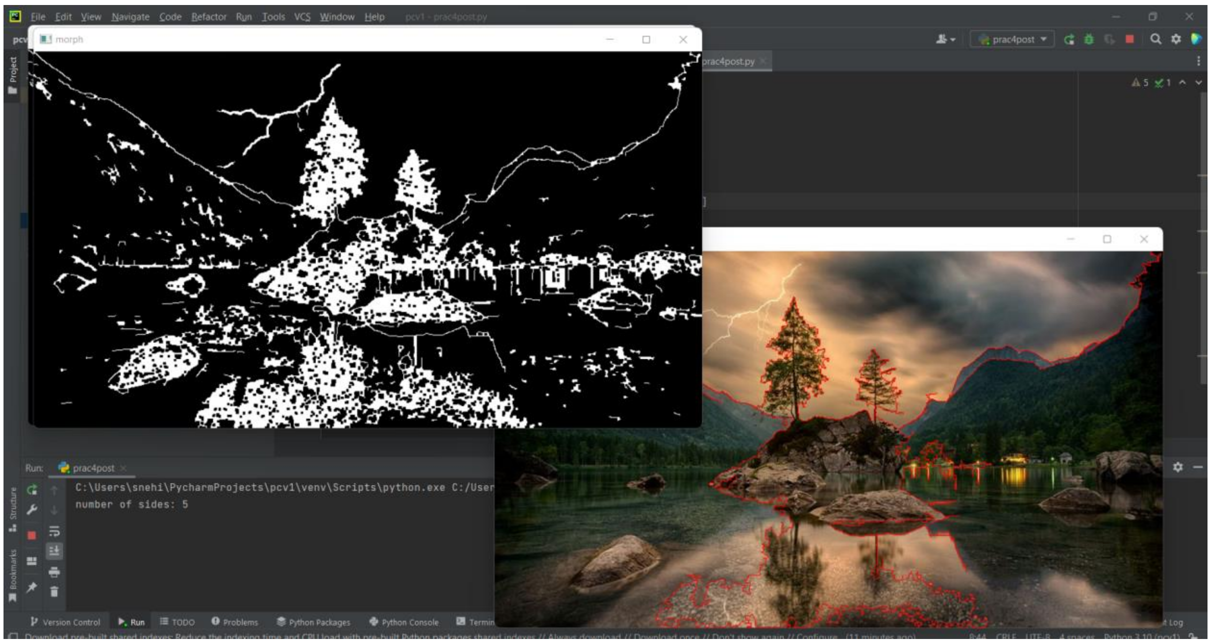
```
[]
alse

lection(event, x, y, flags, param):
ef_point, cropping
 == cv2.EVENT_LBUTTONDOWN:
point = [(x, y)]
ping = True
nt == cv2.EVENT_LBUTTONUP:
point.append((x, y))
ping = False
rectangle(image, ref_point[0], ref_poin
imshow("image", image)

imread(r'C:\Users\snehi\PycharmProjects
e.copy()
dow("image")
Callback("image", shape_selection)
```

```
ehi\PycharmProjects\pcv1\venv\Scripts\py
```

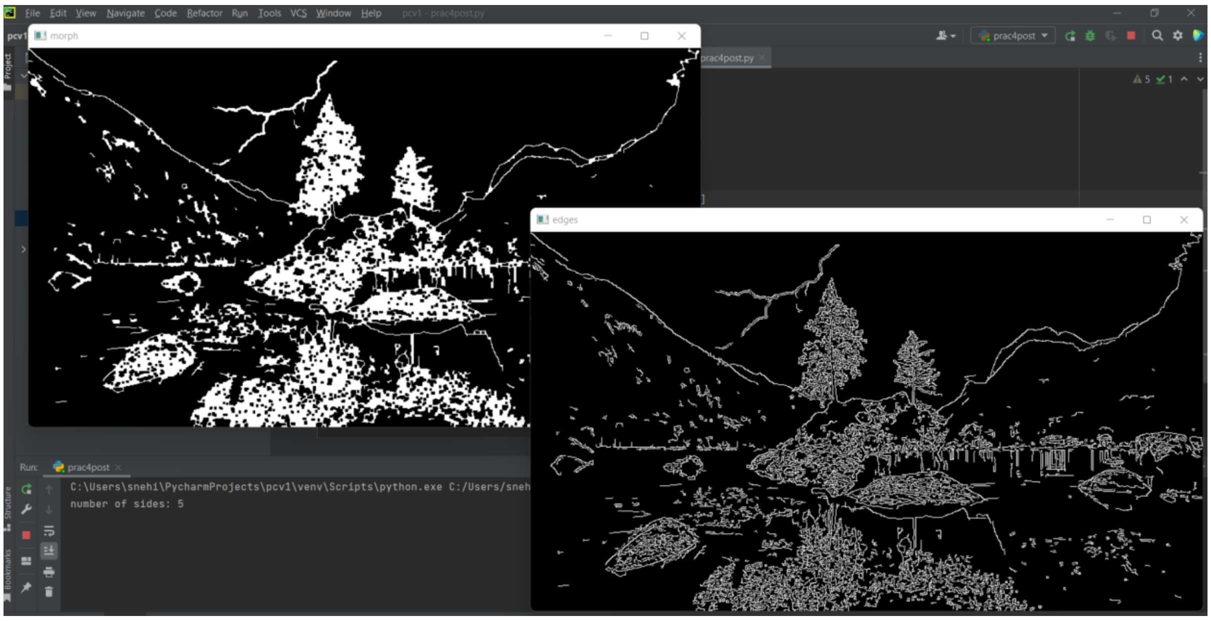
Postlab

1. Charles is a portrait painting artist. He wants to replicate a picture so he wants to know the total number of edges present in the picture. Can you help him find the number of edges in the picture with the help of edge detection

```
import cv2
img = cv2.imread('pcvimage.jpg')
edges = cv2.Canny(img, 100, 200)
kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (3, 3))
morph = cv2.morphologyEx(edges, cv2.MORPH_CLOSE, kernel)
contours = cv2.findContours(morph, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
contours = contours[0] if len(contours) == 2 else contours[1]
big_contour = max(contours, key=cv2.contourArea)
contour = img.copy()
cv2.drawContours(contour, [big_contour], 0, (0,0,255), 1)
peri = cv2.arcLength(big_contour, True)
approx = cv2.approxPolyDP(big_contour, 0.03 * peri, True)
print('number of sides:',len(approx))
cv2.imshow("edges", edges)
cv2.imshow("morph", morph)
cv2.imshow("contour", contour)
cv2.waitKey(0)
cv2.destroyAllWindows()
cv2.destroyAllWindows()
```



```
C:\Users\snehi\PycharmProjects\pcv1\venv\Scripts\python.exe C:/User
number of sides: 5
```


File Edit View Navigate Code Refactor Run Tools VCS Window Help pcv1 - prac4post.py
morph
edges
prac4post.py
prac4post
Run: prac4post

```
C:\Users\snehi\PycharmProjects\pcv1\venv\Scripts\python.exe C:/Users/sneh
number of sides: 5
```