PCV LAB - 5

ID:- 190030059

Name:- A.R.Snehita

1. Mention the different types of thresholding techniques?

A. Simple Thresholding The basic Thresholding technique is Binary Thresholding. For every pixel, the same threshold value is applied. If the pixel value is smaller than the threshold, it is set to 0, otherwise, it is set to a maximum value. The different Simple Thresholding Techniques are: cv2.THRESH_BINARY: If pixel intensity is greater than the set threshold, value set to 255, else set to 0 (black).

cv2.THRESH_BINARY_INV: Inverted or Opposite case of cv2.THRESH_BINARY. cv.THRESH_TRUNC: If pixel intensity value is greater than threshold, it is truncated to the threshold. The pixel values are set to be the same as the threshold. All other values remain the same.

cv.THRESH_TOZERO: Pixel intensity is set to 0, for all the pixels intensity, less than the threshold value.

cv.THRESH_TOZERO_INV: Inverted or Opposite case of cv2.THRESH_TOZERO. Adaptive Thresholding In Simple Thresholding, a global value of threshold was used which remained constant throughout. So, a constant threshold value won't help in the case of variable lighting conditions in different areas. Adaptive thresholding is the method where the threshold value is calculated for smaller regions. This leads to different threshold values for different regions with respect to the change in lighting. We use cv2.adaptiveThreshold for this.

cv2.ADAPTIVE_THRESH_MEAN_C: Threshold Value = (Mean of the neighbourhood area values – constant value). In other words, it is the mean of the blockSize×blockSize neighborhood of a point minus constant.

cv2.ADAPTIVE_THRESH_GAUSSIAN_C: Threshold Value = (Gaussian-weighted sum of the neighbourhood values – constant value). In other words, it is a weighted sum of the blockSize×blockSize neighborhood of a point minus constant.

2. Mention the role of gui in real world?

A. graphical user interface (GUI), a computer program that enables a person to communicate with a computer through the use of symbols, visual metaphors, and pointing devices. Macintosh and Microsoft Corporation’s Windows operating system, the GUI has replaced the arcane and difficult textual interfaces of earlier computing with a relatively intuitive system that has made computer operation not only easier to learn but more pleasant and natural. The GUI is now the standard computer interface, and its components have themselves become unmistakable cultural artifacts.

3. Why is adaptive thresholding used in practice?

A. In Simple Thresholding, a global value of threshold was used which remained constant throughout. So, a constant threshold value won’t help in the case of variable lighting conditions in different areas. Adaptive thresholding is the method where the threshold value is calculated for smaller regions. This leads to different threshold values for different regions with respect to the change in lighting. So we use adaptive thresholding for this.

4. Attribute the reasons for using otsu thresholding.

A. Otsu’s approach processes image histogram, segmenting the objects by minimization of the variance on each of the classes. Usually, this technique produces the appropriate results for bimodal images. The histogram of such image contains two clearly expressed peaks, which represent different ranges of intensity values. The core idea is separating the image histogram into two clusters with a threshold defined as a result of minimization the weighted variance of these classes denoted by \sigma^{2}_w(t).

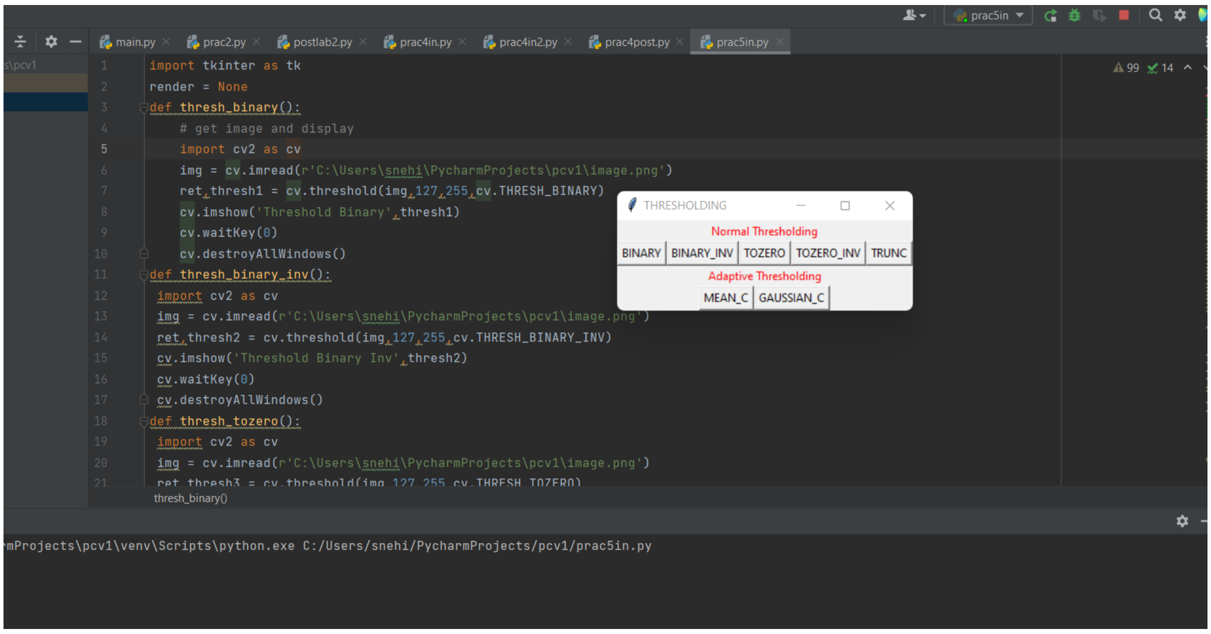
INLAB :-

1)Create a Graphical User Interface using Tkinter as given below and this interface must take an image '.jpg or .png file' and it should perform Thresholding and Adaptive Thresholding .so Create Various buttons like BINARY, BINARY_INV,TOZERO,TOZERO_INV,TRUNC,MEAN_C and GAUSSIAN_C. Here after pressing the button the algorithm must be implemented on the image and give resultant image on the new window

```
import tkinter as tk
render = None
def thresh_binary():
    # get image and display
    import cv2 as cv
    img = cv.imread(r'C:\Users\snehi\PycharmProjects\pcv1\image.png')
    ret,thresh1 = cv.threshold(img,127,255,cv.THRESH_BINARY)
    cv.imshow('Threshold Binary',thresh1)
    cv.waitKey(0)
    cv.destroyAllWindows()
def thresh_binary_inv():
 import cv2 as cv
 img = cv.imread(r'C:\Users\snehi\PycharmProjects\pcv1\image.png')
 ret,thresh2 = cv.threshold(img,127,255,cv.THRESH_BINARY_INV)
 cv.imshow('Threshold Binary Inv',thresh2)
 cv.waitKey(0)
 cv.destroyAllWindows()
def thresh_tozero():
 import cv2 as cv
 img = cv.imread(r'C:\Users\snehi\PycharmProjects\pcv1\image.png')
 ret,thresh3 = cv.threshold(img,127,255,cv.THRESH_TOZERO)
 cv.imshow('Threshold Tozero',thresh3)
 cv.waitKey(0)
 cv.destroyAllWindows()
def thresh_tozero_inv():
 import cv2 as cv
 img = cv.imread(r'C:\Users\snehi\PycharmProjects\pcv1\image.png')
 ret,thresh4 = cv.threshold(img,127,255,cv.THRESH_TOZERO_INV)
 cv.imshow('Threshold Tozero Inv',thresh4)
 cv.waitKey(0)
 cv.destroyAllWindows()
def thresh_trunc():
 import cv2 as cv
 img =cv.imread(r'C:\Users\snehi\PycharmProjects\pcv1\image.png')
 ret,thresh5 = cv.threshold(img,127,255,cv.THRESH_TRUNC)
 cv.imshow('Threshold Trunc',thresh5)
 cv.waitKey(0)
 cv.destroyAllWindows()
def mean_c():
 # get image and display
 import cv2 as cv
 img =cv.imread(r'C:\Users\snehi\PycharmProjects\pcv1\image.png')
 img1 = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
 thresh6 = cv.adaptiveThreshold(img1,
255,cv.ADAPTIVE_THRESH_MEAN_C,cv.THRESH_BINARY, 199, 5)
 cv.imshow('Mean C',thresh6)
 cv.waitKey(0)
 cv.destroyAllWindows()
```

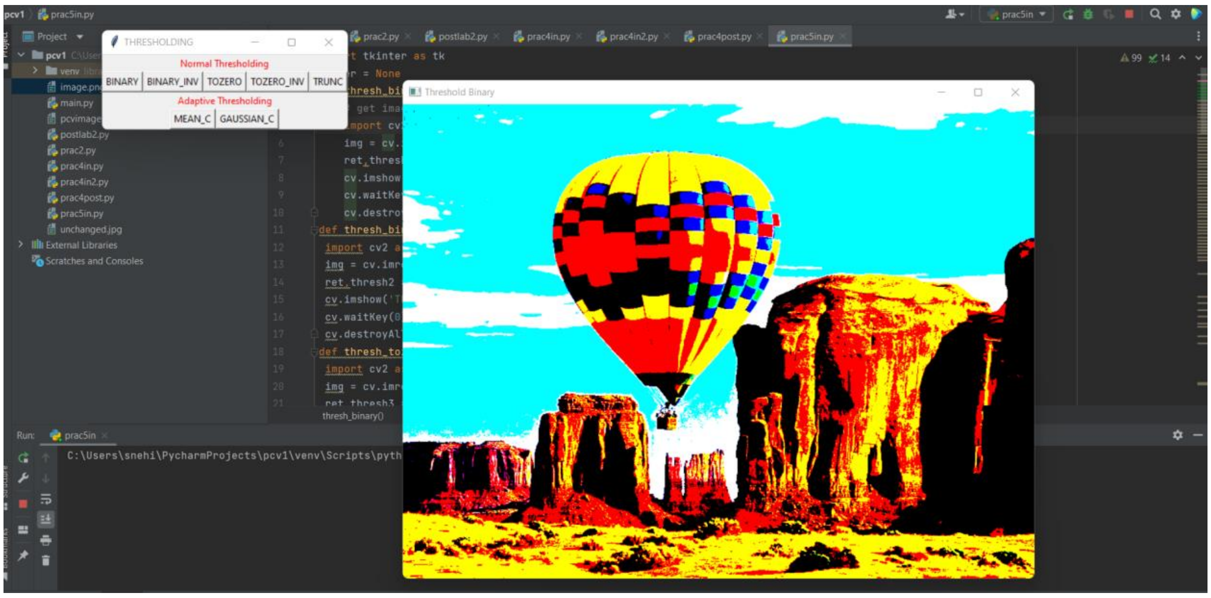
```
def gaussian_c():
 # get image and display
 import cv2 as cv
 img =cv.imread(r'C:\Users\snehi\PycharmProjects\pcv1\image.png')
 img1 = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
 thresh7 = cv.adaptiveThreshold(img1, 255,
cv.ADAPTIVE_THRESH_GAUSSIAN_C,
 cv.THRESH_BINARY, 199, 5)
 cv.imshow('Guassian C',thresh7)
 cv.waitKey(0)
 cv.destroyAllWindows()
root = tk.Tk()
root.title('THRESHOLDING')
frame = tk.Frame(root)
frame.pack()
w = tk.Label(root, text='Normal Thresholding',fg='red')
w.pack()
topFrame = tk.Frame(root)
topFrame.pack()
bottomFrame = tk.Frame(root)
bottomFrame.pack(side=tk.BOTTOM)
button1 = tk.Button(topFrame, text='BINARY',
fg='black',command=thresh_binary)
button2 = tk.Button(topFrame, text='BINARY_INV',
fg='black',command=thresh_binary_inv)
button3 = tk.Button(topFrame, text='TOZERO',
fg='black',command=thresh_tozero)
button4 = tk.Button(topFrame, text='TOZERO_INV',
fg='black',command=thresh_tozero_inv)
button5 = tk.Button(topFrame, text='TRUNC',
fg='black',command=thresh_trunc)
w = tk.Label(root, text='Adaptive Thresholding',fg='red')
w.pack()
button6 = tk.Button(bottomFrame, text='MEAN_C',
fg='black',command=mean_c)
button7 = tk.Button(bottomFrame, text='GAUSSIAN_C',
fg='black',command=gaussian_c)
button1.pack(side=tk.LEFT)
button2.pack(side=tk.LEFT)
button3.pack(side=tk.LEFT)
button4.pack(side=tk.LEFT)
button5.pack(side=tk.LEFT)
button6.pack(side=tk.LEFT)
button7.pack(side=tk.LEFT)
root.mainloop()
```

Screenshots:-

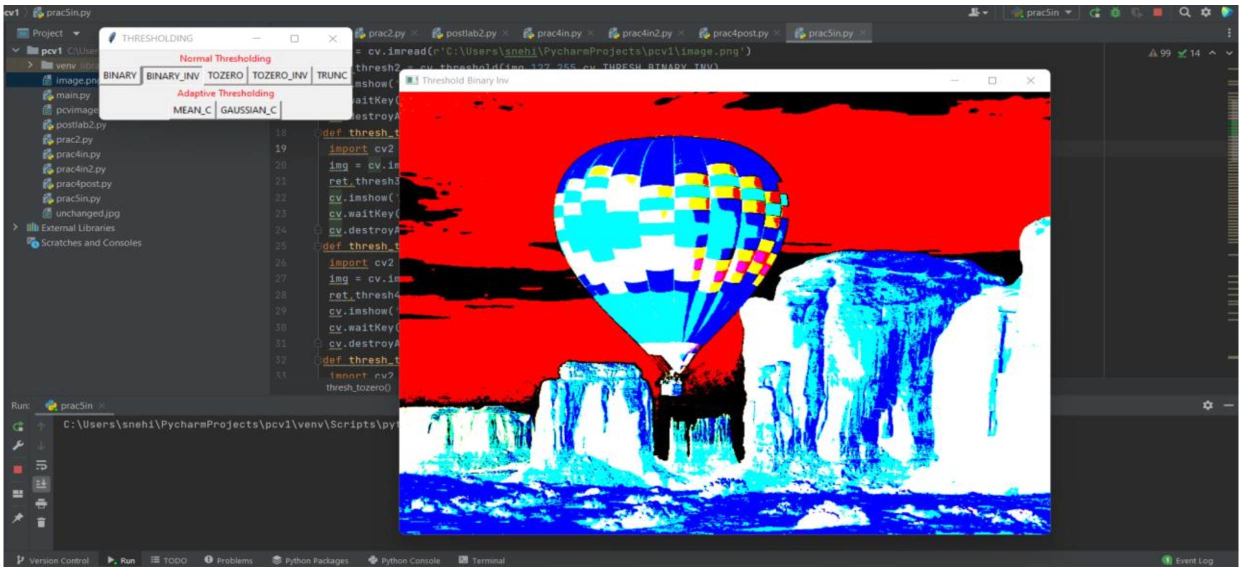


```
import tkinter as tk
render = None
def thresh_binary():
    # get image and display
    import cv2 as cv
    img = cv.imread(r'C:\Users\snehi\PycharmProjects\pcv1\image.png')
    ret,thresh1 = cv.threshold(img,127,255,cv.THRESH_BINARY)
    cv.imshow('Threshold Binary',thresh1)
    cv.waitKey(0)
    cv.destroyAllWindows()
def thresh_binary_inv():
  import cv2 as cv
  img = cv.imread(r'C:\Users\snehi\PycharmProjects\pcv1\image.png')
  ret,thresh2 = cv.threshold(img,127,255,cv.THRESH_BINARY_INV)
  cv.imshow('Threshold Binary Inv',thresh2)
  cv.waitKey(0)
  cv.destroyAllWindows()
def thresh_tozero():
  import cv2 as cv
  img = cv.imread(r'C:\Users\snehi\PycharmProjects\pcv1\image.png')
  ret thresh3 = cv.threshold(img 127 255 cv.THRESH_TOZERO)
```

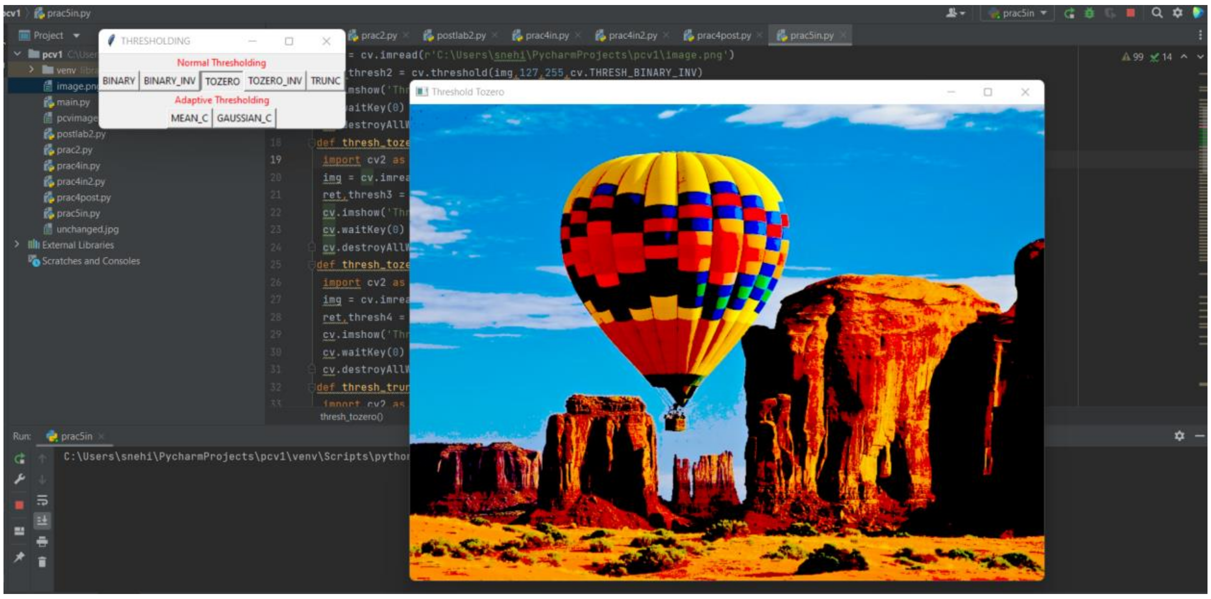
```
mProjects\pcv1\venv\Scripts\python.exe C:/Users/snehi/PycharmProjects/pcv1/prac5in.py
```



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

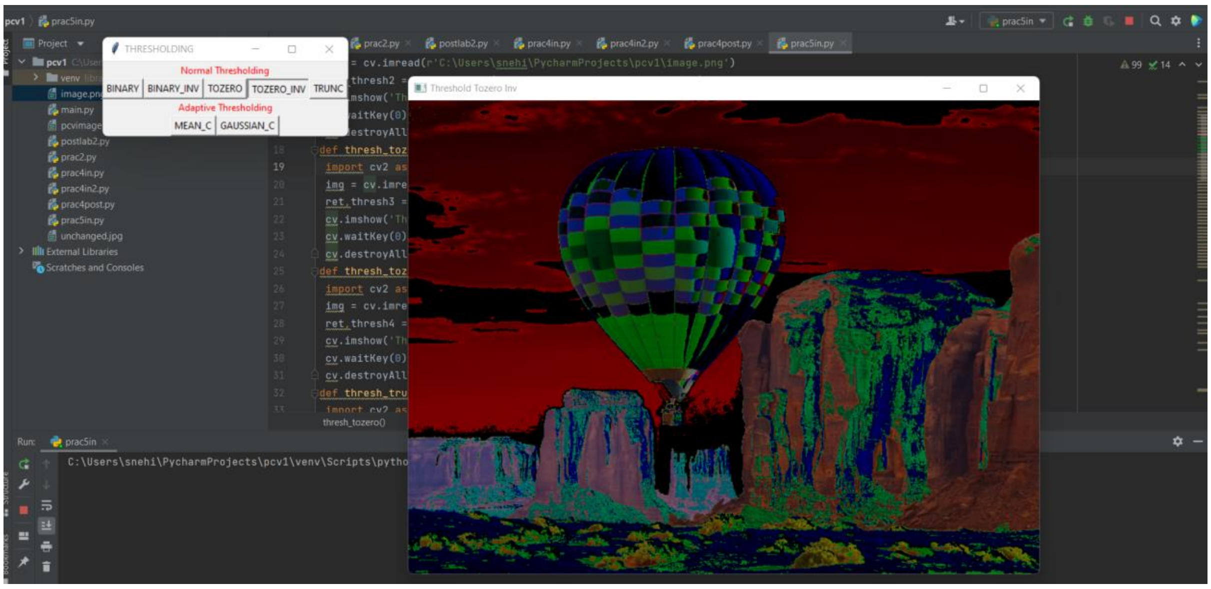


```
C:\Users\snehi\PycharmProjects\pcv1\venv\Scripts\py
```



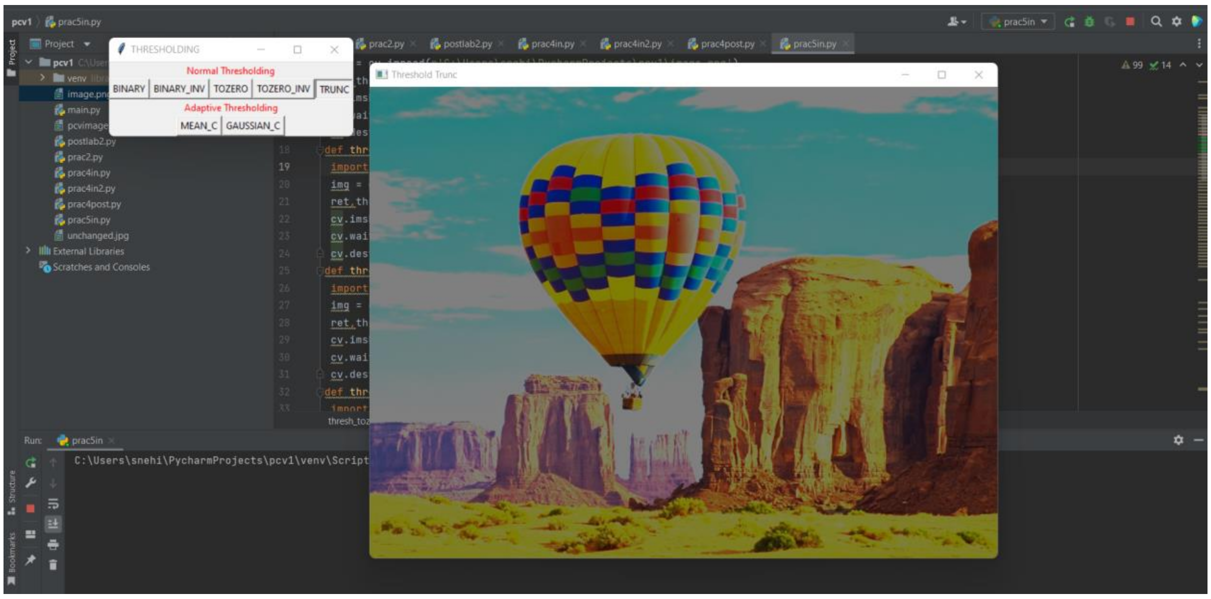
```
= cv.imread(r'C:\Users\snehi\PycharmProjects\pcv1\image.png')
thresh2 = cv.threshold(img,127,255,cv.THRESH_BINARY_INV)
mshow('Th
aitKey(0)
estroyAllW
def thresh_toze
    import cv2 as
    img = cv.imrea
    ret,thresh3 =
    cv.imshow('Th
    cv.waitKey(0)
    cv.destroyAllW
def thresh_toze
    import cv2 as
    img = cv.imrea
    ret,thresh4 =
    cv.imshow('Th
    cv.waitKey(0)
    cv.destroyAllW
def thresh_trun
    import cv2 as
```

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



```
= cv.imread(r'C:\Users\snehi\PycharmProjects\pcv1\image.png')
thresh2 =
mshow('Th
aitKey(0)
estroyAll
def thresh_toz
    import cv2 as
    img = cv.imre
    ret,thresh3 =
    cv.imshow('Th
    cv.waitKey(0)
    cv.destroyAll
def thresh_toz
    import cv2 as
    img = cv.imre
    ret,thresh4 =
    cv.imshow('Th
    cv.waitKey(0)
    cv.destroyAll
def thresh_tru
    import cv2 as
```

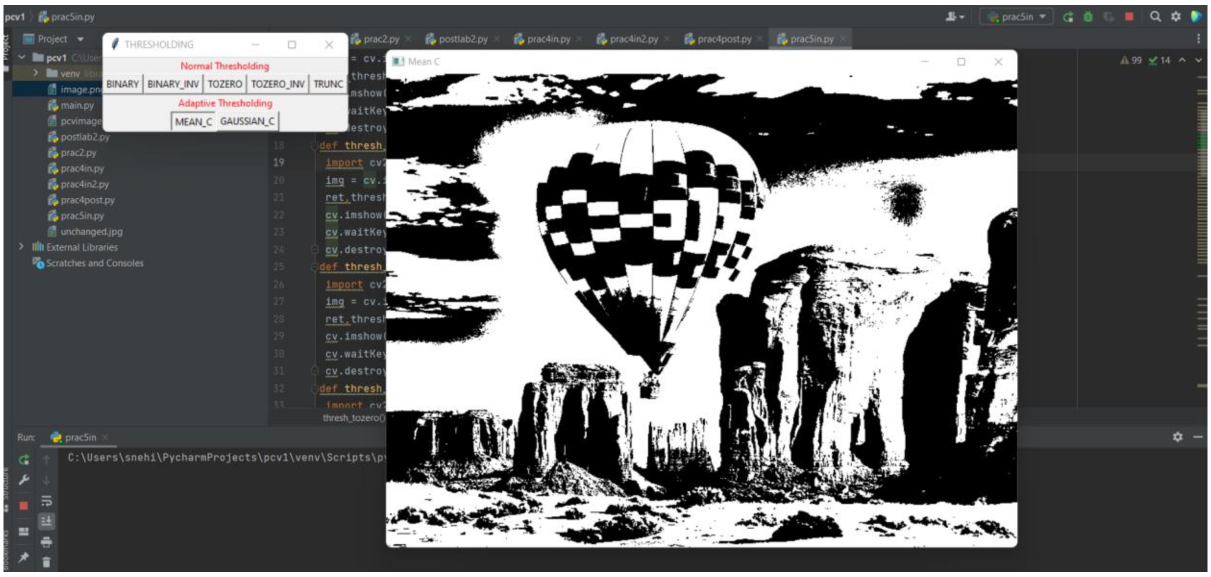
```
C:\Users\snehi\PycharmProjects\pcv1\venv\Scripts\pytho
```



```
def thr
    import
    img =
    ret,th
    cv.ims
    cv.wai
    cv.des
def thr
    import
    img =
    ret,th
    cv.ims
    cv.wai
    cv.des
def thr
    import
```

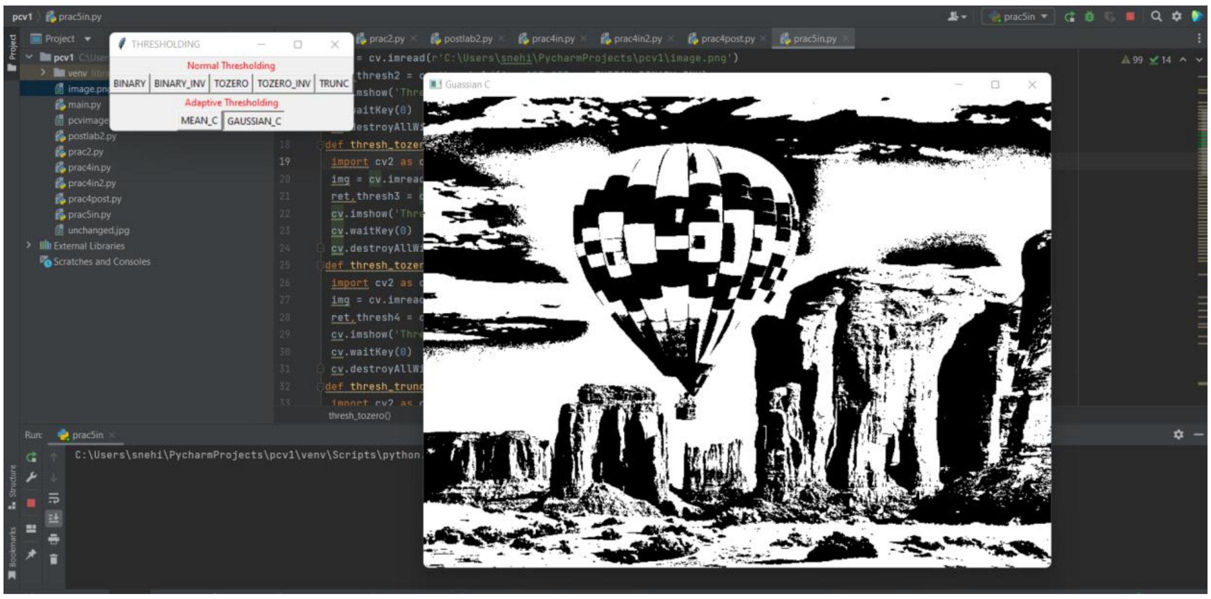
```
C:\Users\snehi\PycharmProjects\pcv1\venv\Script
```

ADAPTIVE THRESHOLDING :-



```
 = cv.i
 thresh
 mshow(
 aitKey
 estroy
def thresh_
    import cv2
    img = cv.i
    ret,thresh
    cv.imshow(
    cv.waitKey
    cv.destroy
def thresh_
    import cv2
    img = cv.i
    ret,thresh
    cv.imshow(
    cv.waitKey
    cv.destroy
def thresh_
    import cv2
```

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

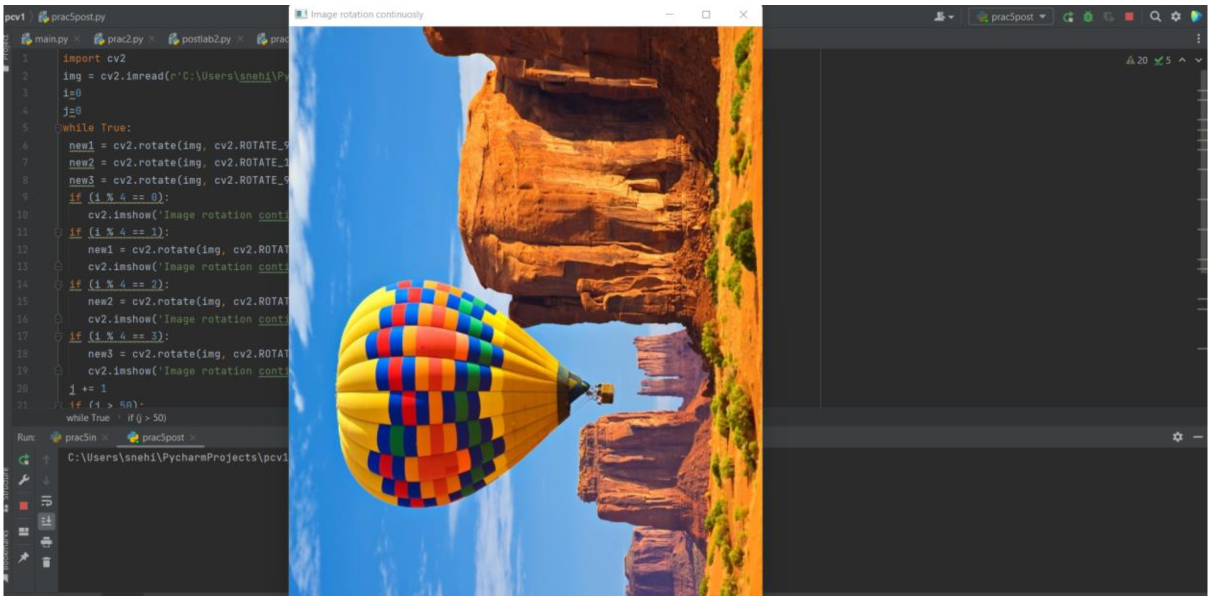


```
 = cv.imread(r'C:\Users\snehi\PycharmProjects\pcv1\image.png')
 thresh2 = c
 mshow('Thre
 aitKey(0)
 estroyAllWi
def thresh_tozer
    import cv2 as c
    img = cv.imread
    ret,thresh3 = c
    cv.imshow('Thre
    cv.waitKey(0)
    cv.destroyAllWi
def thresh_tozer
    import cv2 as c
    img = cv.imread
    ret,thresh4 = c
    cv.imshow('Thre
    cv.waitKey(0)
    cv.destroyAllWi
def thresh_trunc
    import cv2 as c
```

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

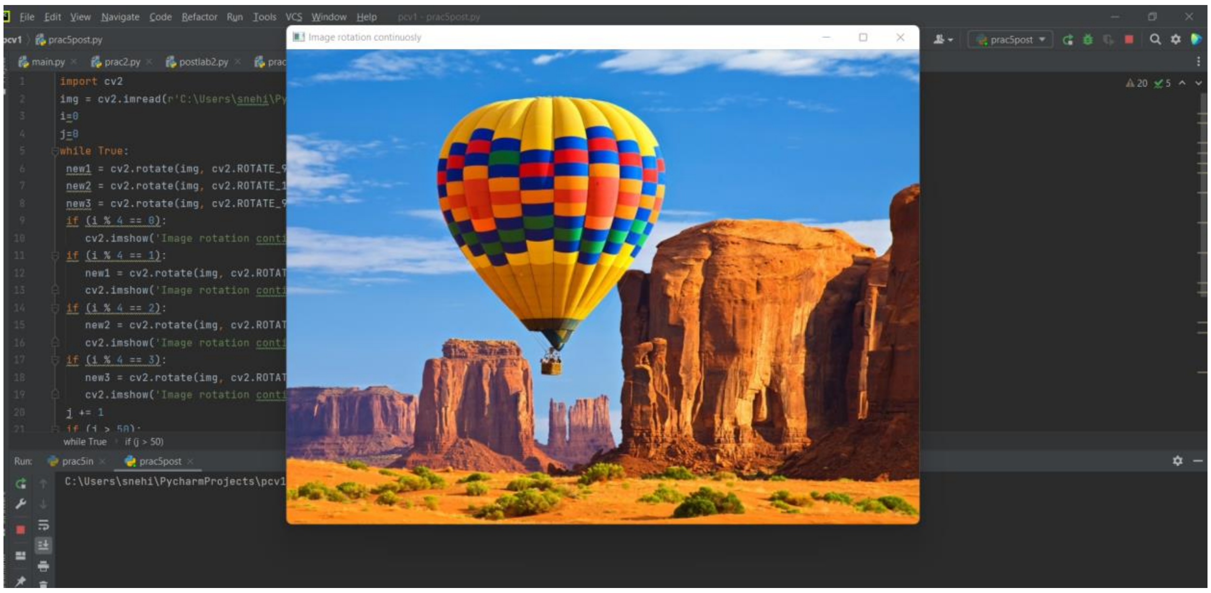
POST LAB:-

1. Jackson is a mathematician guy now he wants to apply tricks on image like he wants to rotate the image along all angles continuously until we press ‘q’ button in keyboard. So help him to perform his task using OPENCV?



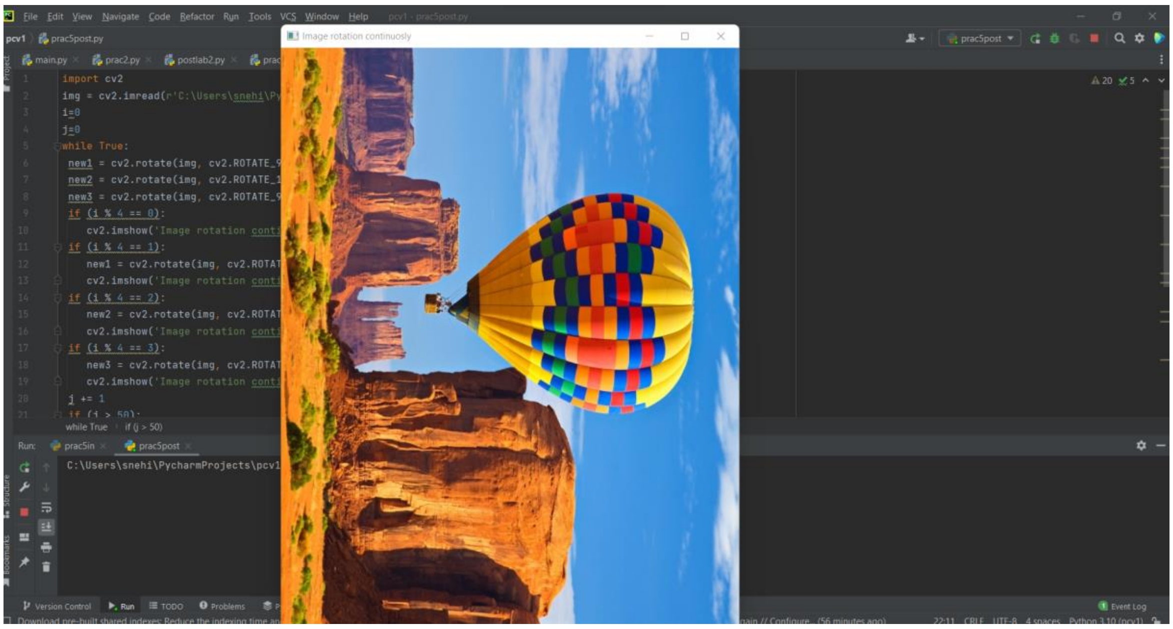
```
import cv2
img = cv2.imread(r'C:\Users\snehi\Py
i=0
j=0
while True:
 new1 = cv2.rotate(img, cv2.ROTATE_9
 new2 = cv2.rotate(img, cv2.ROTATE_1
 new3 = cv2.rotate(img, cv2.ROTATE_9
 if (i % 4 == 0):
     cv2.imshow('Image rotation conti
 if (i % 4 == 1):
     new1 = cv2.rotate(img, cv2.ROTAT
     cv2.imshow('Image rotation conti
 if (i % 4 == 2):
     new2 = cv2.rotate(img, cv2.ROTAT
     cv2.imshow('Image rotation conti
 if (i % 4 == 3):
     new3 = cv2.rotate(img, cv2.ROTAT
     cv2.imshow('Image rotation conti
 j += 1
 if (j > 50):
```

```
C:\Users\snehi\PycharmProjects\pcv1
```



```
import cv2
img = cv2.imread(r'C:\Users\snehi\Py
i=0
j=0
while True:
 new1 = cv2.rotate(img, cv2.ROTATE_9
 new2 = cv2.rotate(img, cv2.ROTATE_1
 new3 = cv2.rotate(img, cv2.ROTATE_9
 if (i % 4 == 0):
     cv2.imshow('Image rotation conti
 if (i % 4 == 1):
     new1 = cv2.rotate(img, cv2.ROTAT
     cv2.imshow('Image rotation conti
 if (i % 4 == 2):
     new2 = cv2.rotate(img, cv2.ROTAT
     cv2.imshow('Image rotation conti
 if (i % 4 == 3):
     new3 = cv2.rotate(img, cv2.ROTAT
     cv2.imshow('Image rotation conti
 j += 1
 if (j > 50):
```

```
C:\Users\snehi\PycharmProjects\pcv1
```



```
import cv2
img = cv2.imread(r'C:\Users\snehi\Py
i=0
j=0
while True:
 new1 = cv2.rotate(img, cv2.ROTATE_9
 new2 = cv2.rotate(img, cv2.ROTATE_1
 new3 = cv2.rotate(img, cv2.ROTATE_9
 if (i % 4 == 0):
     cv2.imshow('Image rotation conti
 if (i % 4 == 1):
     new1 = cv2.rotate(img, cv2.ROTAT
     cv2.imshow('Image rotation conti
 if (i % 4 == 2):
     new2 = cv2.rotate(img, cv2.ROTAT
     cv2.imshow('Image rotation conti
 if (i % 4 == 3):
     new3 = cv2.rotate(img, cv2.ROTAT
     cv2.imshow('Image rotation conti
 j += 1
 if (j > 50):
```

```
C:\Users\snehi\PycharmProjects\pcv1
```

Code:-

```
import cv2
img = cv2.imread(r'C:\Users\snehi\PycharmProjects\pcv1\image.png')
i=0
j=0
while True:
 new1 = cv2.rotate(img, cv2.ROTATE_90_CLOCKWISE)
 new2 = cv2.rotate(img, cv2.ROTATE_180)
 new3 = cv2.rotate(img, cv2.ROTATE_90_COUNTERCLOCKWISE)
 if (i % 4 == 0):
    cv2.imshow('Image rotation continuosly', img)
 if (i % 4 == 1):
    new1 = cv2.rotate(img, cv2.ROTATE_90_CLOCKWISE)
    cv2.imshow('Image rotation continuosly', new1)
 if (i % 4 == 2):
    new2 = cv2.rotate(img, cv2.ROTATE_180)
    cv2.imshow('Image rotation continuosly', new2)
 if (i % 4 == 3):
    new3 = cv2.rotate(img, cv2.ROTATE_90_COUNTERCLOCKWISE)
    cv2.imshow('Image rotation continuosly', new3)
 j += 1
 if (j > 50):
    i += 1
    j = 0
 j+=1
 if cv2.waitKey(1) & 0xFF == ord('q'):
    break

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