PCV LAB - 2

ID:- 190030059

Name:- A.R.Snehita

PRELAB

1. How is an RGB image represented? How many channels are there in an RGB image?

A. A M-by-N RGB image is a 2D matrix where each matrix element is a vector with 3 values. In grayscale images, each pixel can be represented by a single number, which typically ranges from 0 to 255. This value determines how dark the pixel appears (e.g., 0 is black, while 255 is bright white). In colored images, each pixel can be represented by a vector of three numbers (each ranging from 0 to 255) for the three primary color channels: red, green, and blue. These three red, green, and blue (RGB) values are used together to decide the color of that pixel. For example, purple might be represented as 128, 0, 128 (a mix of moderately intense red and blue, with no green). There are three channels in it – Red , Blue and Green

2. What are the ranges of pixel intensities for each of these channels?

A. If the RGB image is 24-bit (the industry standard as of 2005), each channel has 8 bits, for red, green, and blue—in other words, the image is composed of three images (one for each channel), where each image can store discrete pixels with conventional brightness intensities between 0 and 255. If the RGB image is 48-bit (very high color-depth), each channel is made of 16-bit images.

3. What are color-spaces?

A. Color spaces are different types of color modes, used in image processing and signals and system for various purposes. A "color space" is a useful conceptual tool for understanding the color capabilities of a particular device or digital file. When trying to reproduce color on another device, color spaces

can show whether you will be able to retain shadow/highlight detail, color saturation, and by how much either will be compromised.

4. Describe the following color-spaces.

i) YCrCb Color Space

A. Y'CbCr color model contains Y', the luma component and cb and cr are the blue-difference and red difference chroma components. It is not an absolute color space. It is mainly used for digital systems Its common applications include JPEG and MPEG compression. Y'UV is often used as the term for Y'CbCr, however they are totally different formats. The main difference between these two is that the former is analog while the later is digital.

ii) HSV color space

A)HSV is closer to how humans perceive color. It has three components: hue, saturation, and value. This color space describes colors (hue or tint) in terms of their shade (saturation or amount of gray) and their brightness value. Some color pickers, like the one in Adobe Photoshop, use the acronym HSB, which substitutes the term "brightness" for "value," but HSV and HSB refer to the same color model.

iii) LAB color space

A)Lab Color is a more accurate color space. It uses three values (L, a, and b) to specify colors. RGB and CMYK color spaces specify a color by telling a device how much of each color is needed. Lab Color works more like the human eye.It specifies a color using a 3-axis system. The a-axis (green to red), b-axis (blue to yellow) and Lightness axis.

iv) CMYK color space

The CMYK color model is based off of the fact that surfaces appear to be certain colors because of the wavelengths of light they absorb and reflect. For example, things that appear to be red will only reflect red light. Objects that appear to be white will reflect all wavelengths of light, and objects that appear black absorb all wavelengths

v) BGR Color Space

A. RGB is the most widely used color space, and we have already discussed it in the past tutorials. RGB stands for red green and blue. What RGB model states, that each color image is actually formed of three different images. Red image, Blue image, and black image. A normal grayscale image can be defined by only one matrix, but a color image is actually composed of three different matrices. One color image matrix = red matrix + blue matrix + green matrix

vi) Edge map of image

Ans) An edge map is an image that indicated where edges are in the image. The image will have some kind of edge detection filter and the values in the edge map image may reflect the strength of the edge, or it may have already been thresholded (like the edge() function) so that you have a binary image.Usually people want to emphasize edges, not soften them.

vii) Heat map of image

With a potential energy function, we'll use several visualizations to see what it looks like. One important example is called a heatmap, which uses darker and brighter colors to show how the value of a scalar field changes over a 2D space

viii) Spectral Image map

A. The spectral image presents a three-dimensional array of data which combines precise spectral information with two-dimensional spatial correlation. The analysis of a spectral image creates a unique database, which enables the extraction of features and the evaluation of quantities from multipoint spectral information, impossible to obtain otherwise.

INLAB:-

1. Perform the following operations programmatically using OpenCV and python: Comment your code as much as possible.

1. Read an input color (RGB) image

2. Convert the read image to YCrCb color space

3. Convert the read image to HSV color space

4. Convert the read image to LAB color space

5. Compute the edge map of the read image using Laplacian

6. Compute the heat map of the read image

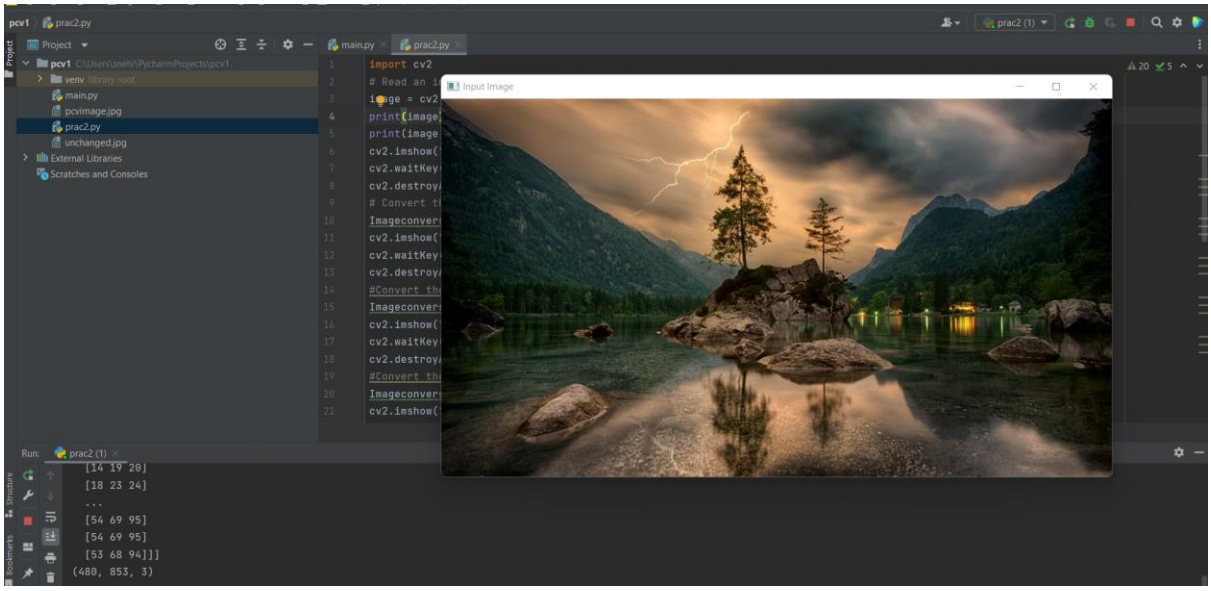
7. Compute the spectral image map of the read image

Code:-

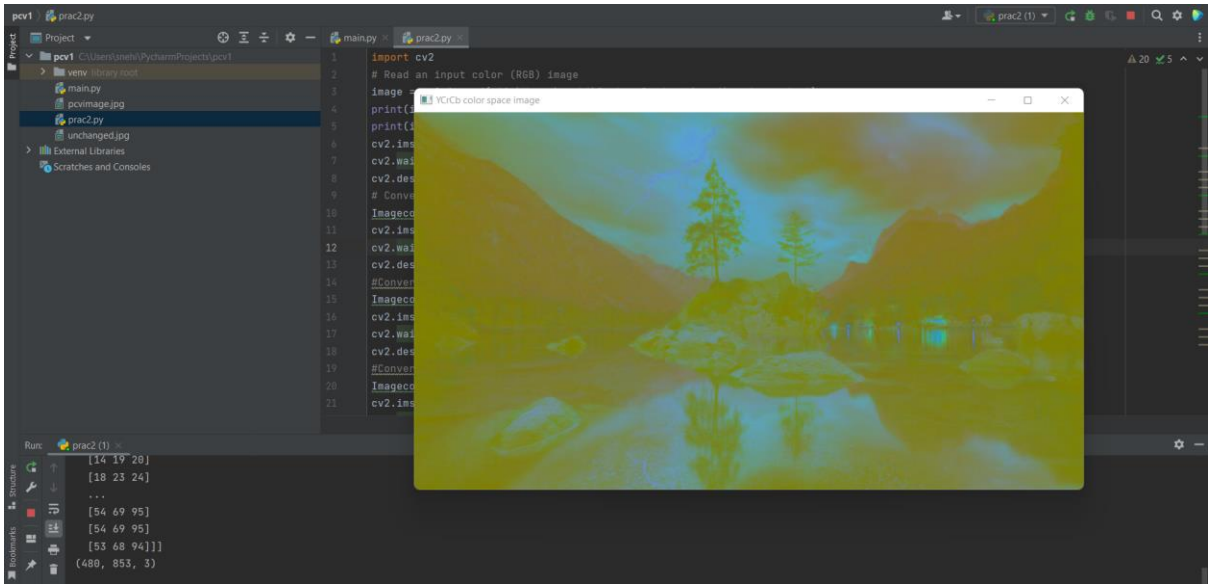
```
import cv2
# Read an input color (RGB) image
image =
cv2.imread(r'C:\Users\snehi\PycharmProjects\pcv1\pcvimage.jpg'
)
print(image)
print(image.shape)
cv2.imshow("Input Image", image)
cv2.waitKey(0)
cv2.destroyAllWindows()
# Convert the read image to YCrCb color space
Imageconversion = cv2.cvtColor(image, cv2.COLOR_BGR2YCR_CB)
cv2.imshow("YCrCb color space image",Imageconversion)
cv2.waitKey(0)
cv2.destroyAllWindows()
#Convert the read image to HSV color space
Imageconversion1=cv2.cvtColor(image,cv2.COLOR_BGR2HSV)
cv2.imshow("HSV color space image",Imageconversion1)
cv2.waitKey(0)
cv2.destroyAllWindows()
#Convert the read image to LAB color space
Imageconversion2=cv2.cvtColor(image,cv2.COLOR_BGR2LAB)
cv2.imshow("LAB color space image",Imageconversion2)
cv2.waitKey(0)
cv2.destroyAllWindows()
#Compute the edge map of the read image using Laplacian
Image1=cv2.Laplacian(image,cv2.CV_64F)
cv2.imshow("Edge Map using Laplacian",Image1)
cv2.waitKey(0)
cv2.destroyAllWindows()
#Compute the heat map of the read image
Image2=cv2.applyColorMap(image,cv2.COLORMAP_HOT)
cv2.imshow("Heat Map",Image2)
cv2.waitKey(0)
cv2.destroyAllWindows()
#Compute the spectral image map of the read image
from matplotlib import pyplot as plt
plt.imshow(image, cmap ='nipy_spectral')
plt.title("Spectral color space image")
plt.show()
```

Screenshots:

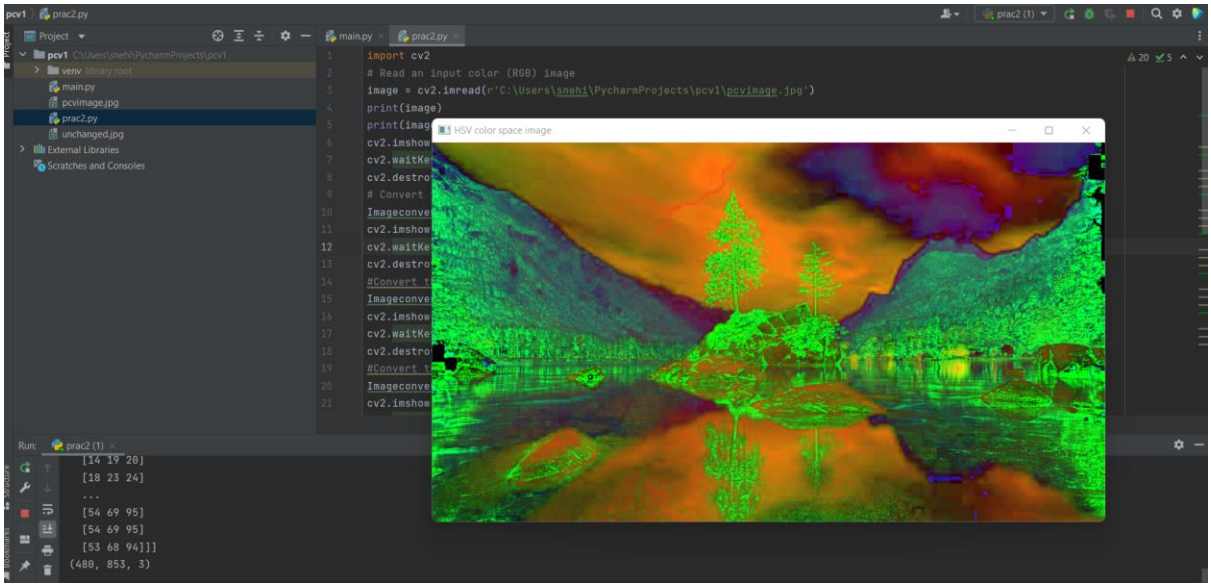
1) Read an input color (RGB) image



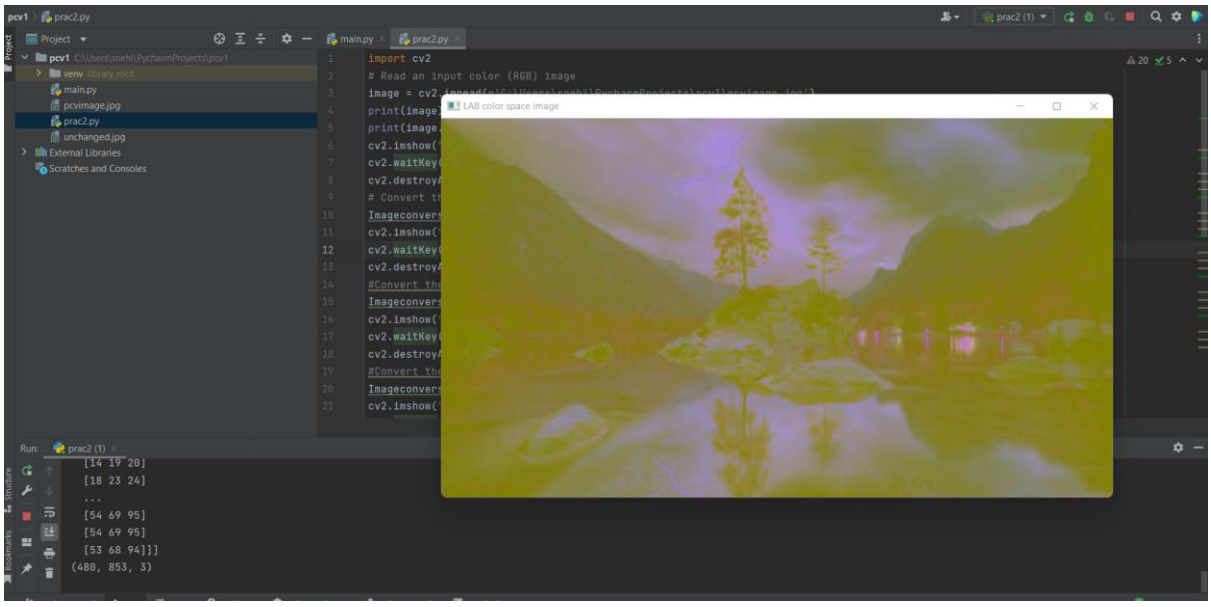
2) Convert the read image to YCrCb color space



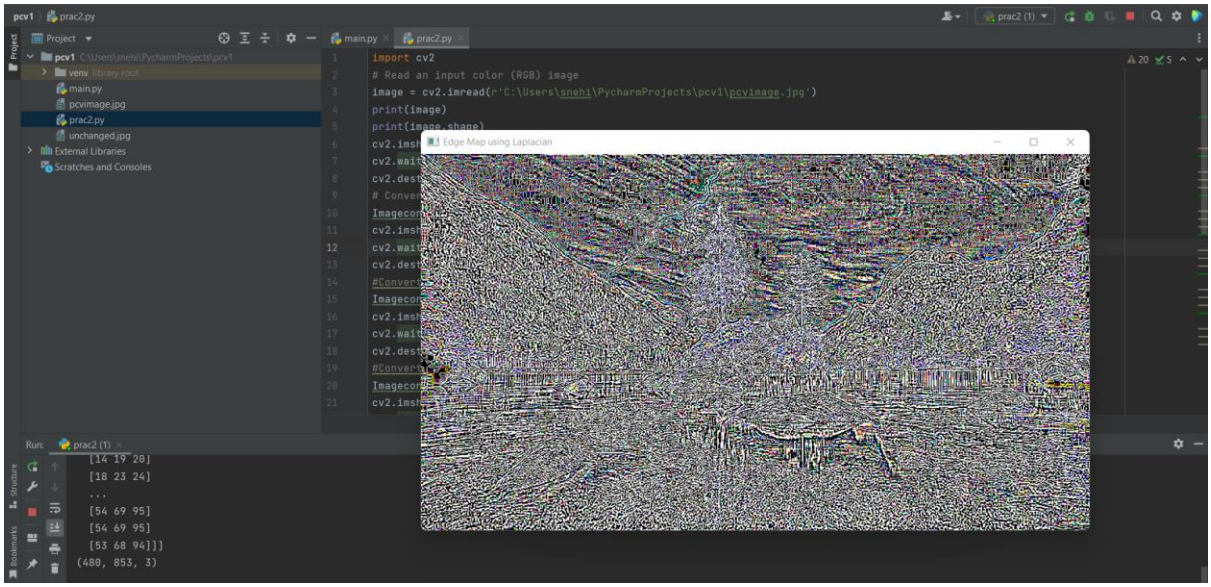
3) Convert the read image to HSV color space



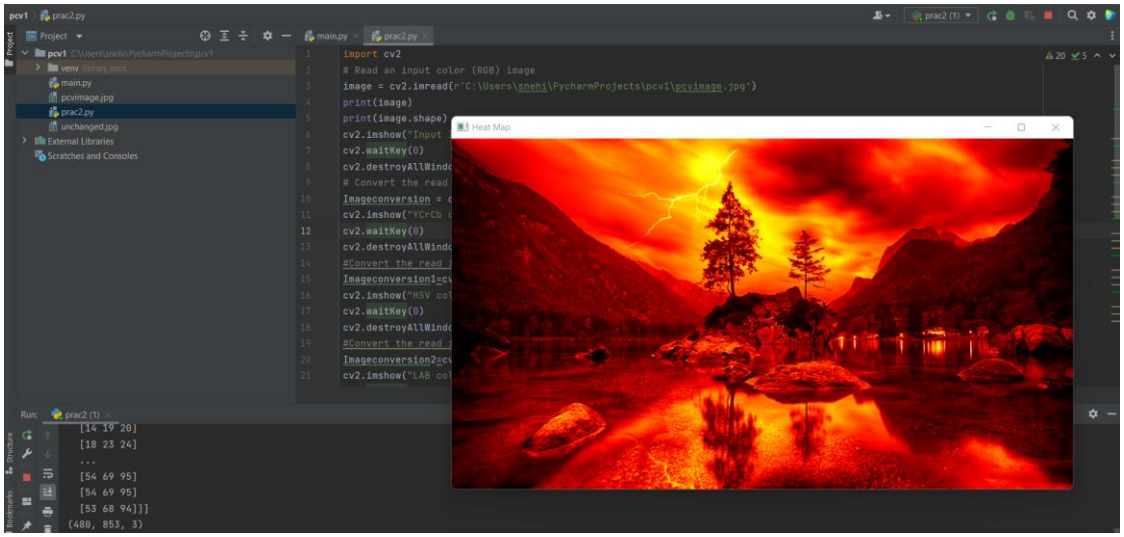
4. Convert the read image to LAB color space



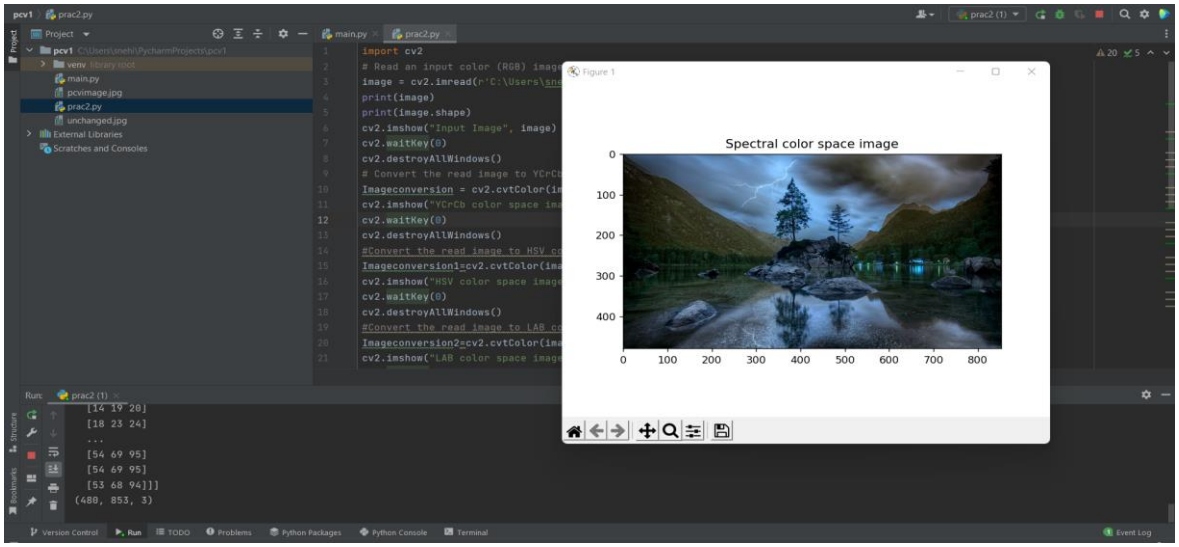
5. Compute the edge map of the read image using Laplacian



6. Compute the heat map of the read image

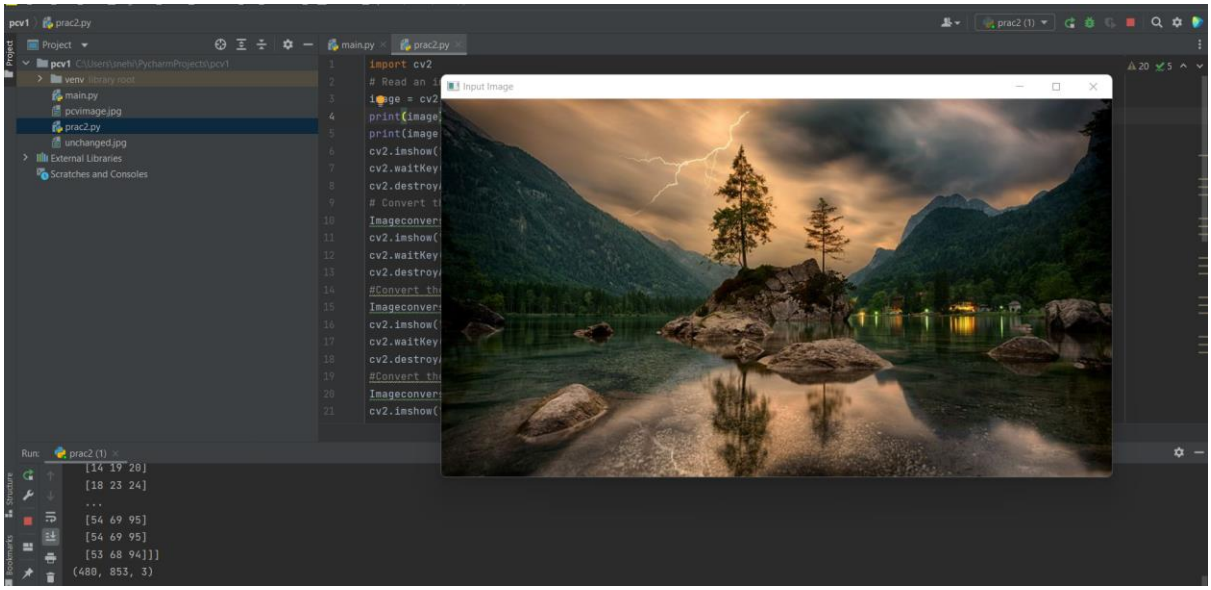


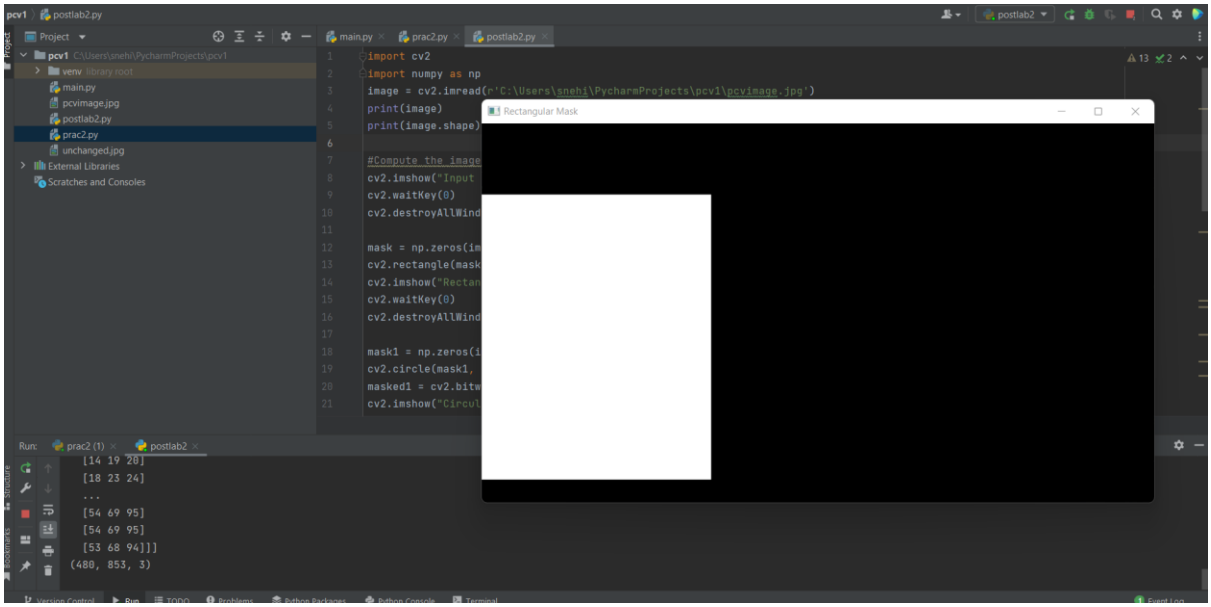
7. Compute the spectral image map of the read image

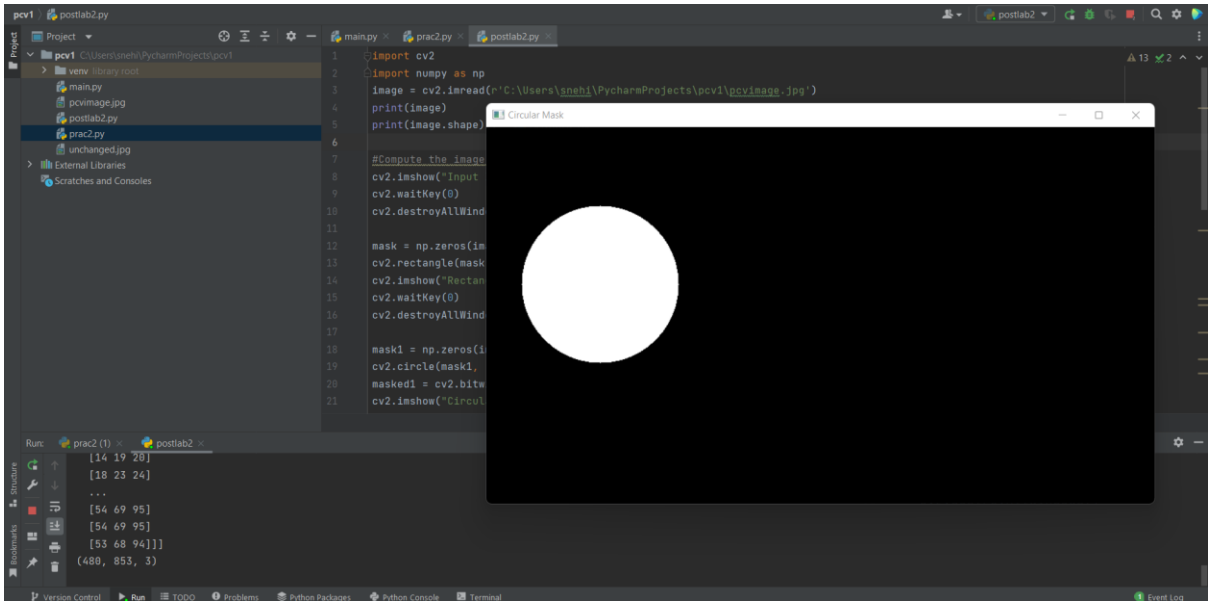


Postlab

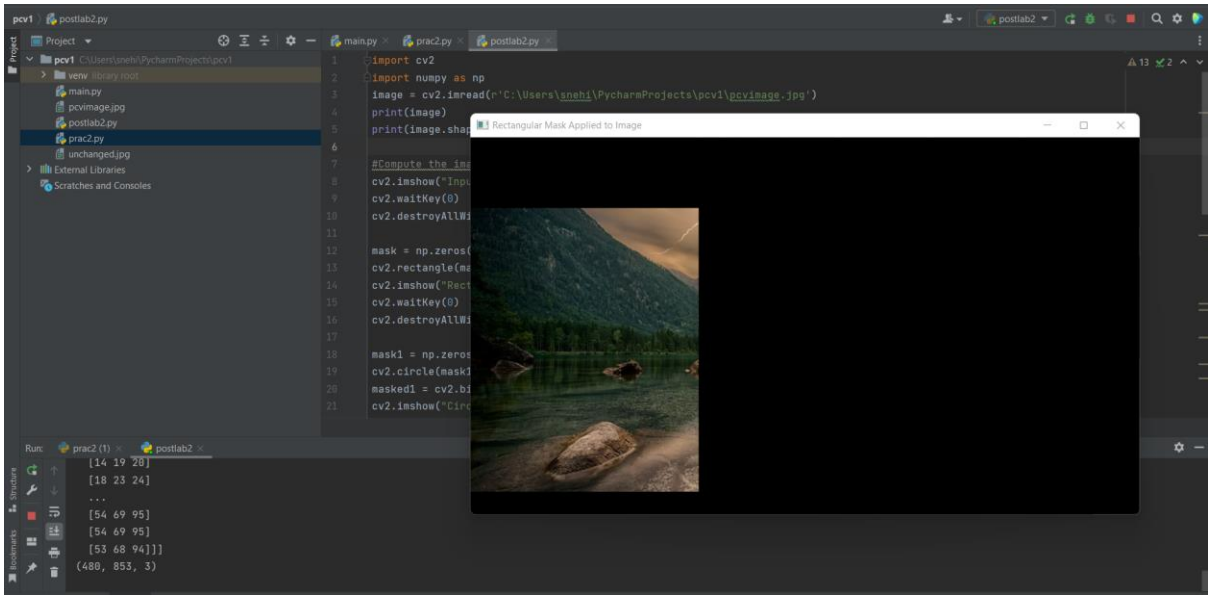
1. Compute the image mask of the input (read) image you used in the IN-LAB TASK

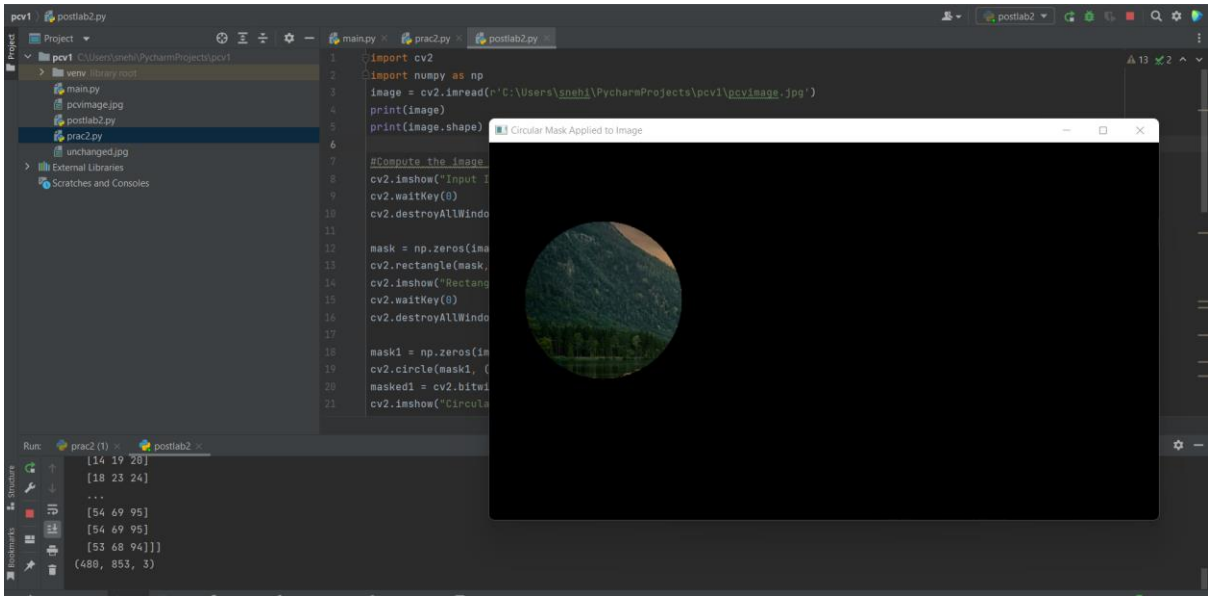




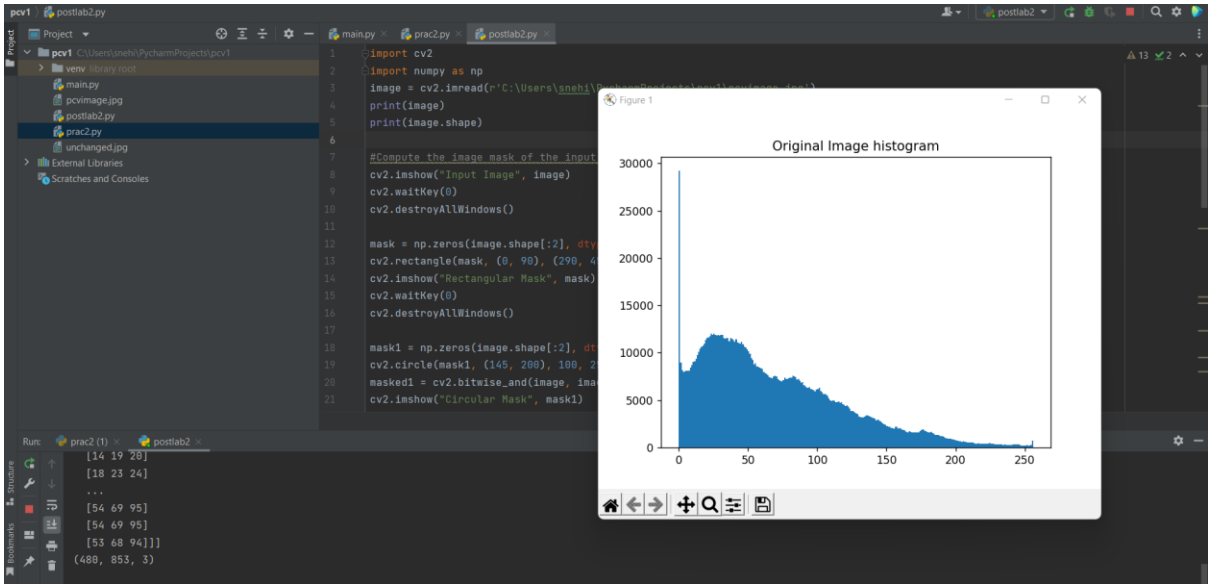


2. Now superimpose the color image on top of the mask image and display the result





3. Plot the resulting histograms of the original color input image, the computed mask image and the superimposed (mask+color) images



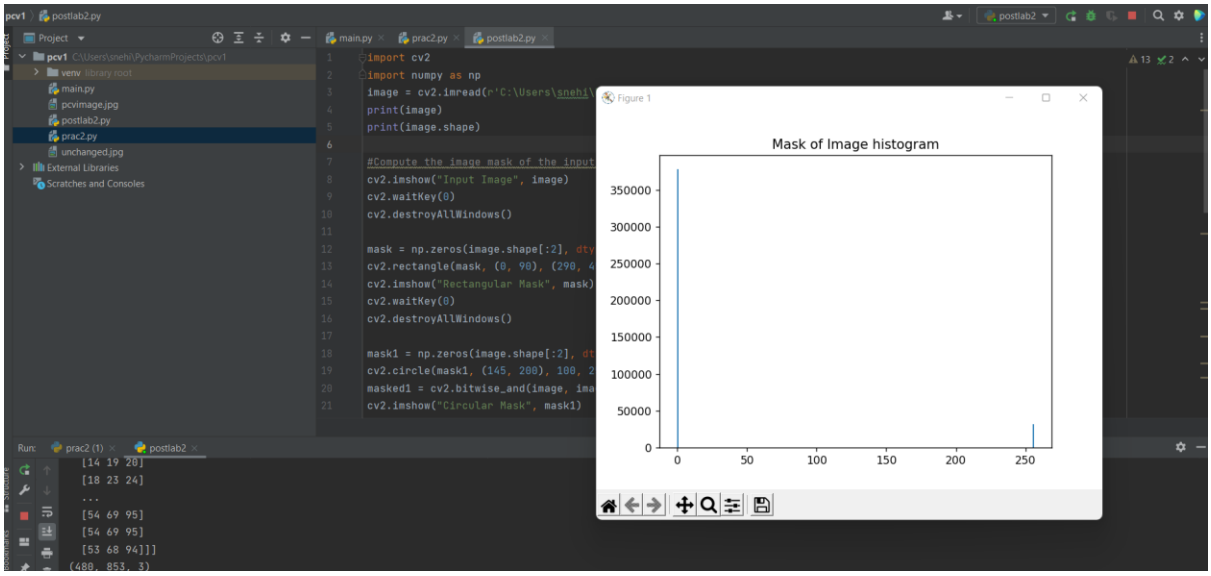


Figure 1
Mask of Image histogram
350000
300000
250000
200000
150000
100000
50000
0
0
50
100
150
200
250

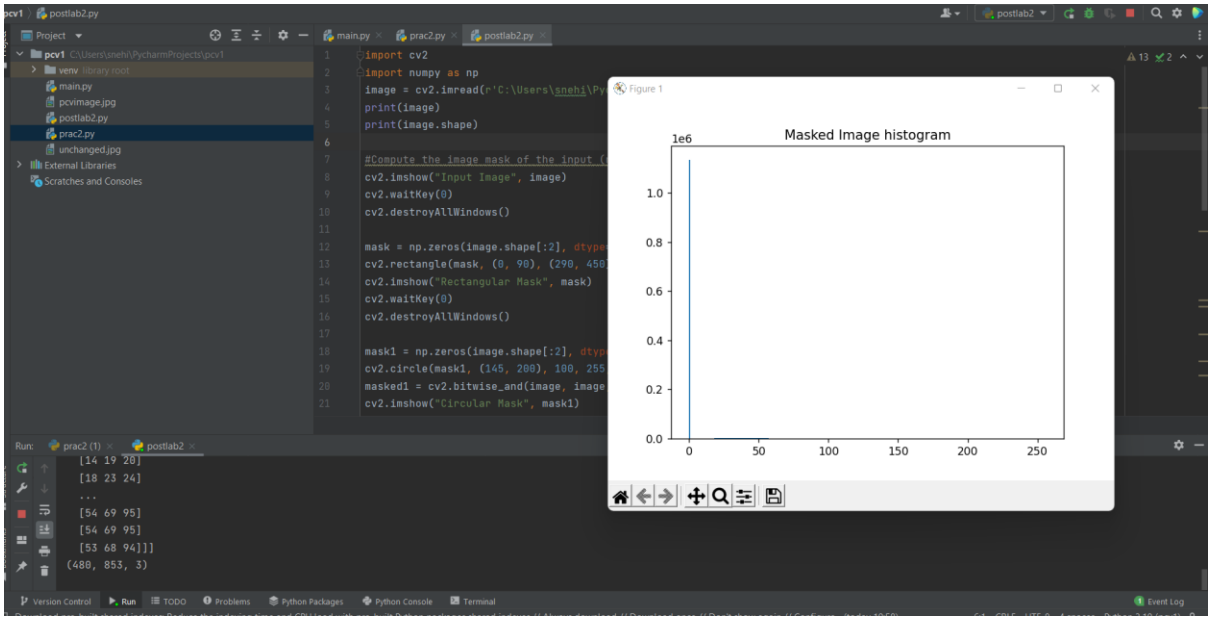


Figure 1
Masked Image histogram
1e6
1.0
0.8
0.6
0.4
0.2
0.0
0
50
100
150
200
250