Home Work-0

CMSC 733-0101: Computer Processing of Pictorial Information,Fall-2022

1.Plot the R, G, B values along the scanline on the 250th row of the image.

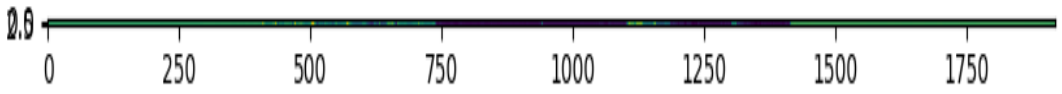
Programming Language: Python

Software: PyCharm

Code:

```
# Import the necessary libraries

from PIL import Image

from numpy import asarray

import matplotlib.pyplot as plt

img = Image.open('iribefront.jpg')

# asarray() class is used to convert PIL images into NumPy arrays

pic= asarray(img)

new_image=pic[250,:,[0,1,2]]

plt.figure(figsize = (10,10))

plt.imshow(new_image)

plt.show()

#plt.savefig('1_scanline.png')
```

Output of problem number:01



2.Stack the R, G, B channels of the image vertically.

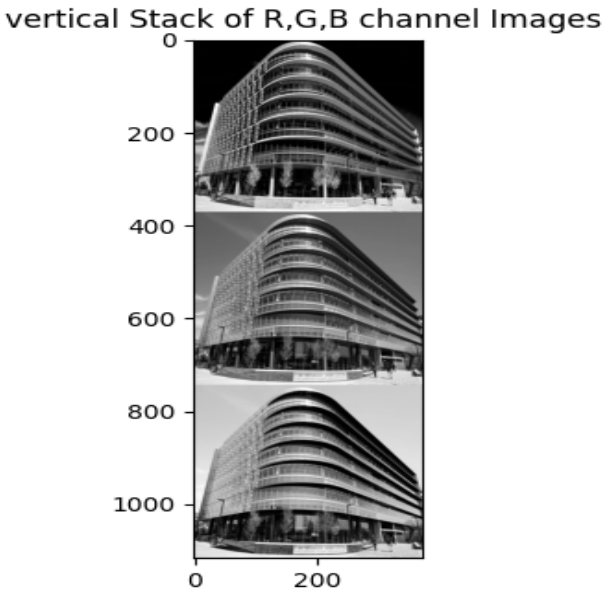
Programming Language: Python

Software: PyCharm

Code:

```
from PIL import Image

import matplotlib.pyplot as plt

img = Image.open('iribefront.jpg')

#resizing image

img1 = img.resize((372,372))

# Split methods that splits images into 3 different channels

pic = Image.Image.split(img1)

#creating a new bank image to make stack

new_image = Image.new("RGB", (372,1116), "white")

# pasting the R channel Image to new blank image

new_image.paste(pic[0], (0, 0))

# pasting the G channel Image to new blank image

new_image.paste(pic[1], (0, 372))

# pasting the B channel image and position

new_image.paste(pic[2], (0,744))

# pasting the second image and position

plt.title('vertical Stack of R,G,B channel Images ')

plt.imshow(new_image)

plt.show()
```

Output of Problem 2:



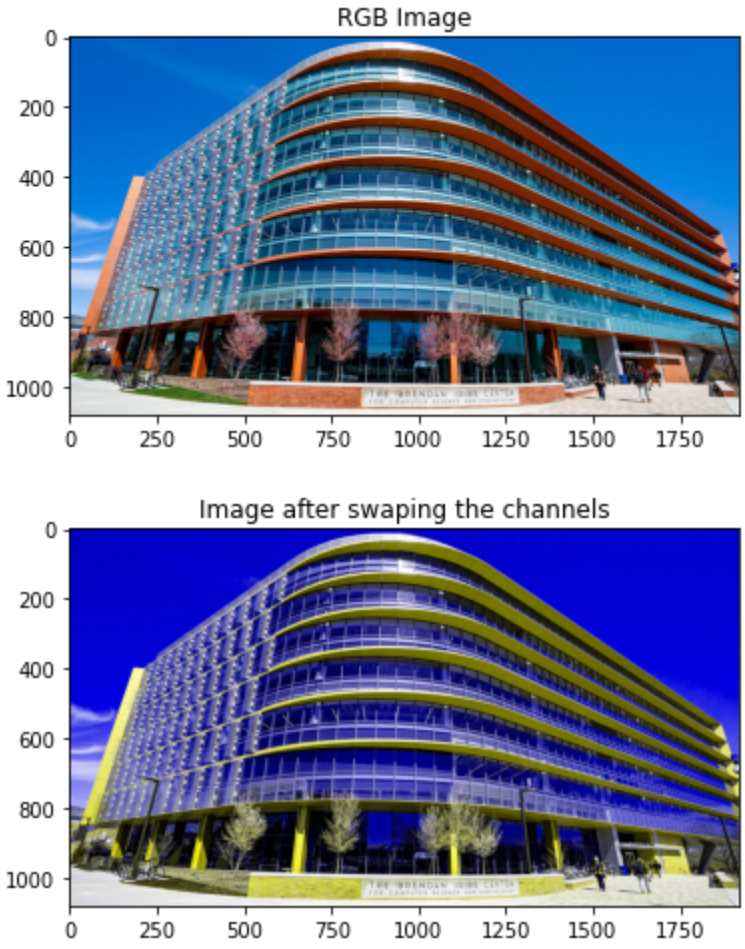
3.Load the input color image and swap its red and green color channels.

Code:

```
import numpy as np

import imageio

import matplotlib.pyplot as plt

%matplotlib inline

pic = imageio.imread('https://iribe.umd.edu/img/iribefront.jpg')

plt.title('RGB Image')

plt.imshow(pic)

plt.show()
```

```
plt.title('Image after swapping the channels')

pic[:,:,1]=pic[:,:,0]

pic[:,:,0]=pic[:,:,1]

#pic[:,:,0]=pic[:,:,0]

plt.imshow(pic)

plt.show()
```

Output of problem 3

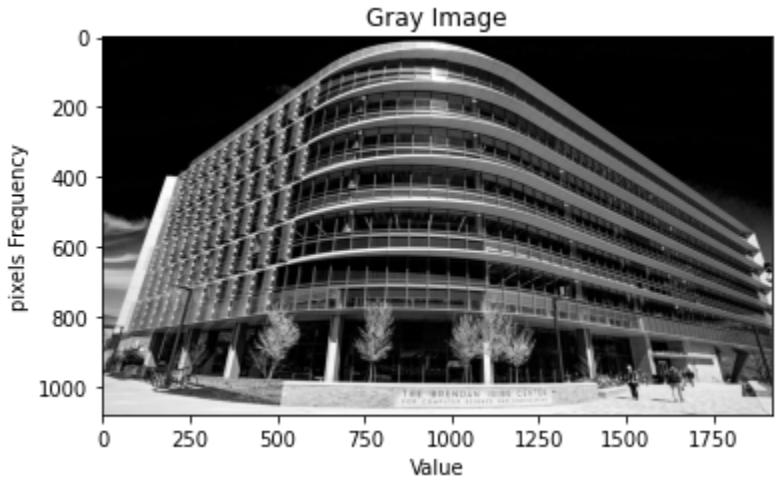


4.Convert the input color image to a grayscale image.

Code:

```
import imageio

from PIL import Image

import matplotlib.pyplot as plt

import matplotlib.image as mpimg

from PIL import ImageFilter

import numpy as np

img= imageio.imread('https://iribe.umd.edu/img/iribefront.jpg')

R, G, B = img[:,:,0], img[:,:,1], img[:,:,2]

rgb2gray = 0.299 * R + 0.587 * G + 0.114 * B

plt.imshow(rgb2gray, cmap='gray')

x=img[:,:,0]

plt.xlabel("Value")

plt.ylabel("pixels Frequency")

plt.title("Gray Image")

plt.imshow(x,cmap="gray")

plt.show()

plt.savefig('Gray image')
```

Output of Problem4



5.Take the R, G, B channels of the image. Compute an average over the three channels. Note that you may need to do the necessary typecasting (uint8 and double) to avoid overflow.

Code:

```
import matplotlib.pyplot as plt

from PIL import Image

import numpy as np

from PIL import Image, ImageStat

pic = Image.open('iribefront.jpg')

avg = ImageStat.Stat(pic)

print(avg.mean)
```

Output of problem5:

```
"C:\Users\R K\PycharmProjects\pythonProject\venv\Scripts\python.exe" "C:/Users/R K/PycharmProjects/pythonProject/5_average.py"
[73.20855902777778, 105.51530333719136, 134.04692515432097]

Process finished with exit code 0
```

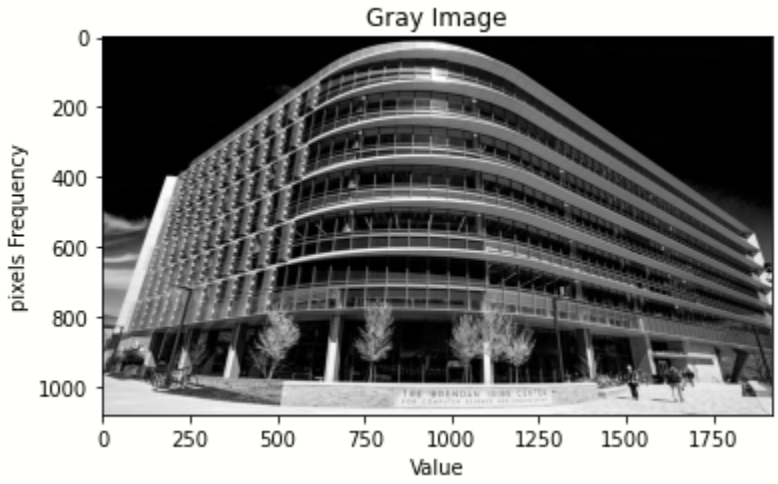
6.Take the grayscale image in (4), obtain the negative image (i.e., mapping 255 to 0 and 0 to 255).

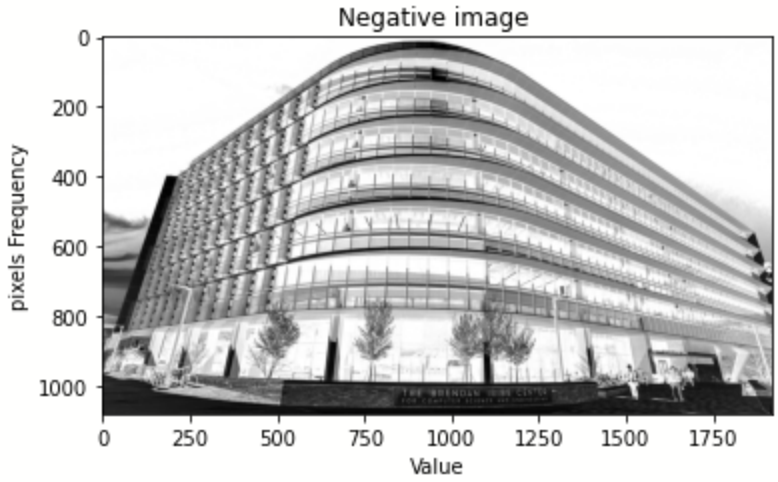
Code:

```
import imageio

from PIL import Image

import matplotlib.pyplot as plt

import matplotlib.image as mpimg

from PIL import ImageFilter

import numpy as np

img= imageio.imread('https://iribe.umd.edu/img/iribefront.jpg')

R, G, B = img[:,:,0], img[:,:,1], img[:,:,2]

#RGB to gray conversion

rgb2gray = 0.299 * R + 0.587 * G + 0.114 * B

plt.imshow(rgb2gray, cmap='gray')

x=img[:,:,0]

plt.xlabel("Value")

plt.ylabel("pixels Frequency")

# title of an image .

plt.title("Gray Image")

plt.imshow(x,cmap="gray")

#Image plotting

plt.show()

plt.savefig('Gray image')
```

```
y=np.shape(x)

z=np.zeros(y)

#negative Image conversion

z=255-x

plt.xlabel("Value")

plt.ylabel("pixels Frequency")

plt.title("Negative image ")

plt.imshow(z,cmap="gray")

plt.show()
```

Output of Problem6





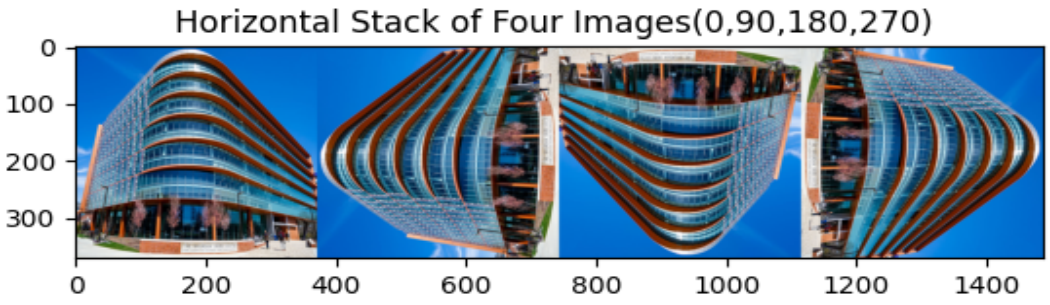
7. First, crop the original image into a squared image of size 372 x 372. Then, rotate the image by 90, 180, and 270 degrees and stack the four images (0, 90, 180, 270 degreess) horizontally.

Code:

```
import matplotlib.pyplot as plt

import numpy as np

from PIL import Image

img = Image.open('iribefront.jpg')

img1 = img.resize((372,372))

img2 = img1.rotate(90)

img3 = img1.rotate(180)

img4 = img1.rotate(270)

#img4.show()

# creating a new image and pasting the

# images

img5 = Image.new("RGB", (1488, 372), "white")

# pasting the img1
```

```
img5.paste(img1, (0, 0))

# pasting the second image and position

img5.paste(img2, (372, 0))

# pasting the second image and position

img5.paste(img3, (744, 0))

# pasting the second image and position

img5.paste(img4, (1116, 0))

plt.title('Horizontal Stack of Four Images(0,90,180,270) ')

plt.imshow(img5)

plt.show()
```

Output of problem7



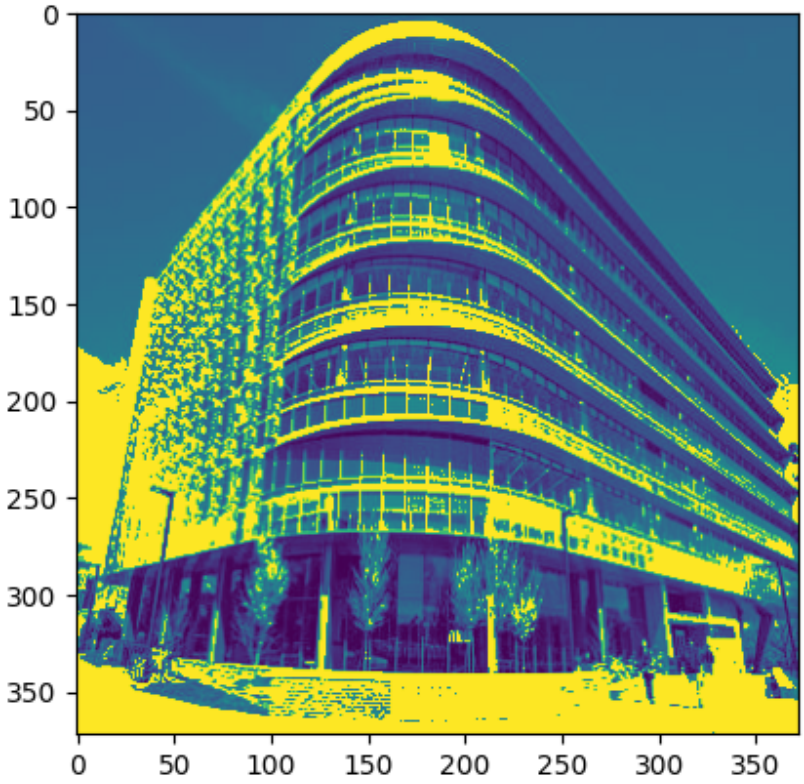
8. Create another image with the same size as the image. First, initialize this image as zero everywhere. Then, for each channel, set the pixel values as 255 when the corresponding pixel values in the image are greater than 127.

Code:

```
from PIL import Image, ImageStat

import matplotlib.pyplot as plt

import numpy as np

from PIL import Image

import PIL

img = Image.open('iribefront.jpg')

#make the image as the same size of the previous image

img1 = img.resize((372,372))

img1=img1.save("convert.jpg")

im2 = Image.new(mode="RGB", size=(372,372),color="black")

im2=im2.save("black.jpg")

mask=Image.new(mode="RGB", size=(372,372),color="white")

mask=mask.save("mask.jpg")

im1=Image.open('convert.jpg').convert('L')

im2= Image.open('black.jpg').convert('L')

mask= Image.open('mask.jpg').convert('L')

im3 = PIL.Image.composite(im1, im2, mask)

im3.show()

#convert the image into numpy array

pic = np.array(im3)

pic2 = np.where(pic>127,255,pic)
```

```
plt.imshow(pic2)

plt.show()
```

Output of Problem8



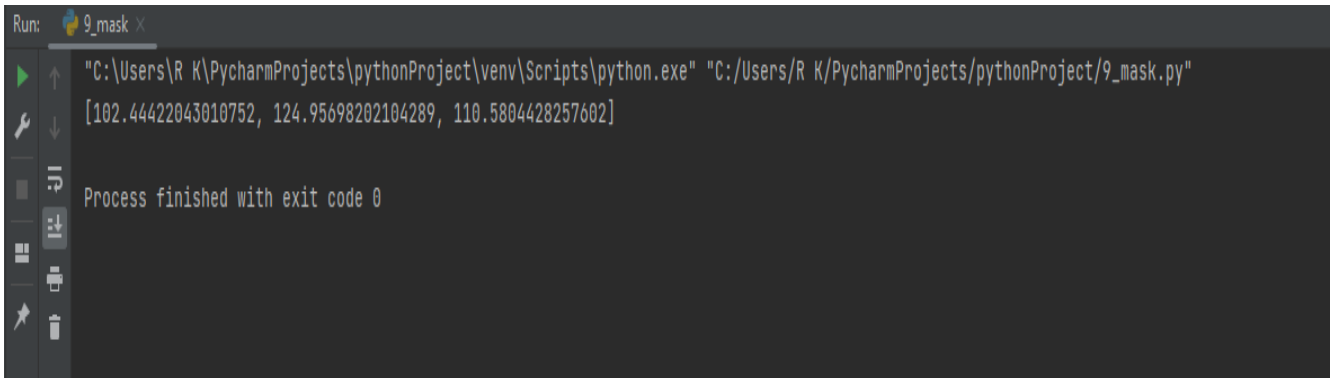
9. Report the mean R, G, B values for those pixels marked by the mask in (8)

Code:

```
from PIL import Image, ImageStat

import matplotlib.pyplot as plt

import numpy as np

from PIL import Image

import PIL
```

```
img = Image.open('iribefront.jpg')
#make the image as the same size of the previous image
img1 = img.resize((372,372))
img1=img1.save("convert.jpg")
im2 = Image.new(mode="RGB", size=(372,372),color="black")
im2=im2.save("black.jpg")
mask=Image.new(mode="RGB", size=(372,372),color="white")
mask=mask.save("mask.jpg")
im1=Image.open('convert.jpg').convert('L')
im2= Image.open('black.jpg').convert('L')
mask= Image.open('mask.jpg').convert('L')
im3 = PIL.Image.composite(im1, im2, mask)
im3.show()
#convert the image into numpy array
pic = np.array(im3)
pic2 = np.where(pic>127,255,pic)
plt.imshow(pic2)
plt.show()
#avg calculation.
plt.imsave('finl_mask_image.jpg', pic2)
pic3=Image.open('finl_mask_image.jpg')
avg = ImageStat.Stat(pic3)
print(avg.mean)
```

Output of Problem9



10.Take the grayscale image in (3). Create and initialize another image as all zeros. For each 5 x 5 window in the grayscale image, find out the maximum value and set the pixels with the maximum value in the 5x5 window as 255 in the new image

Code:

```
from numpy import asarray

import matplotlib.pyplot as plt

from scipy.signal import convolve2d as conv2

from PIL import Image

import matplotlib.pyplot as plt

import numpy as np

pic = Image.open('iribefront.jpg')

#convert the image into a numpy array

img=np.array(pic)

R, G, B = img[:,:,0], img[:,:,1], img[:,:,2]

#Gray Scale Conversion

rgb2gray = 0.299 * R + 0.587 * G + 0.114 * B
```

```
# imshow function with comparison of gray level value.
output = conv2(rgb2gray,np.ones((5,5),dtype=int), 'same') == np.amax(rgb2gray)
rgb2gray[output] = 255
plt.xlabel("Value")
plt.ylabel("pixels Frequency")
plt.title("New Image")
plt.imshow(rgb2gray)
plt.show()
```

Output of Problem10