PAMANTASAN NG LUNGSOD NG MAYNILA
(University of the City of Manila)
Intramuros, Manila

Elective 3

Laboratory Activity No. 2
Image Representation, Color Models, and Image Operations

Score

Submitted by:
Cacal, Chad R.
Dela Cruz, Alfonso Rafael C.
Palabay, Joven Carl B.
Suyu, Chester T.
SAT 7:00AM – 4:00PM / CPE 0332.1-1

Date Submitted
30-07-2024

Submitted to:
Engr. Maria Rizette H. Sayo

I. Objectives

This laboratory activity aims to implement the principles and techniques of image acquisition, representation, color models through MATLAB/Octave and open CV using Python

1. Acquire the image.
2. Acquire image representation.
3. Acquire image color models.
4. Modify image representation.
5. Flip Image.

II. Methods

A. Perform a task given in the presentation

- Copy and paste your MATLAB code

```
% Read an image
img = imread('E:\PLM CET SUBJECTS\Digital Image Processing\flower.jpg');

% Display the image
figure(1);
imshow(img); title('Original Image');

% Get image dimensions (rows, columns, color channels)[rows, cols, channels]
= size(img);
disp(['Image size: ', num2str(rows), ' x ', num2str(cols), ' x ',num2str(channels)]);

% Check color model (grayscale or RGB)if
channels == 1
    disp('Color Model: Grayscale');else
    disp('Color Model: RGB');end

% Access individual pixels (example: center pixel)center_row = floor(rows/2) +
1;
center_col = floor(cols/2) + 1;
center_pixel = img(center_row, center_col, :); disp(['Center pixel value: ',
num2str(center_pixel)]);

% Basic arithmetic operations (add constant value to all pixels)brightened_img = img + 50;
figure (2);
imshow(brightened_img); title ('Image Brightened');
% Basic geometric operation (flipping image horizontally)flipped_img =
fliplr(img);
figure(3);
imshow(flipped_img); title('Image Flipped Horizontally');
```

B. Supplementary Activity

- Write a Python program that will implement the output in Method A.

```
import cv2
import numpy as np
#Read an image
img = cv2.imread('flower.jpg')
#Display the image
cv2.imshow('Original Image', img)
#Get image dimensions (rows, columns, color channels)
[rows,cols,channels] = img.shape
print("Image size: " + str(rows) + " x " + str(cols) + " x " +
str(channels))
#Check color model (grayscale or RGB)
if channels == 1:
    print('Color Model: Grayscale')
else:
    print('Color Model: RGB')
#Access individual pixels(example: center pixel)
center_row = rows//2 + 1
center_col = cols//2 + 1
center_pixel = img[center_row,center_col,:]
print("Center pixel: ", end="")
index = 2
while index >= 0:
    print(center_pixel[index],end=" ")
    index-=1
#Basic arithmetic operations(add constant value to all pixels)
brightness = np.ones(img.shape, dtype="uint8")*50
brightened_img = cv2.add(img,brightness)
cv2.imshow('Image Brightened', brightened_img)
#Basic geometric operation (flipping image horizontally)
flipped_img = cv2.flip(img, 1)
cv2.imshow('Image Flipped Horizontally', flipped_img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

III. Results

Image Attribute and Color Model

- Image size: 1536 x 1536 x 3
- Color model: RGB
- Center pixel value: 91 109 109

Steps:

1. Copy/crop and paste your results. Label each output (Figure1, Figure2, Figure3)

picture file: flower.jpg

MATLAB Results

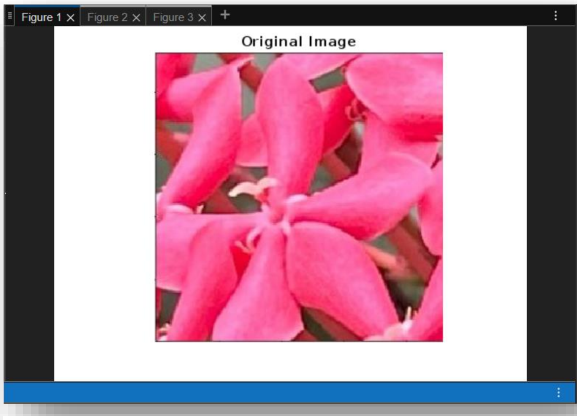

Figure 1. Acquiring an Image of a Flower in MATLAB

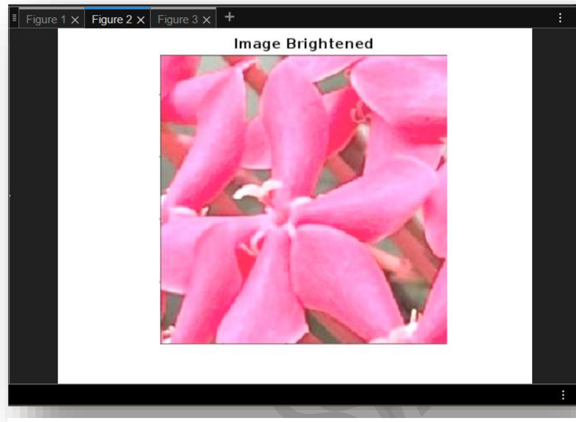

Figure 2. Image brightened in MATLAB

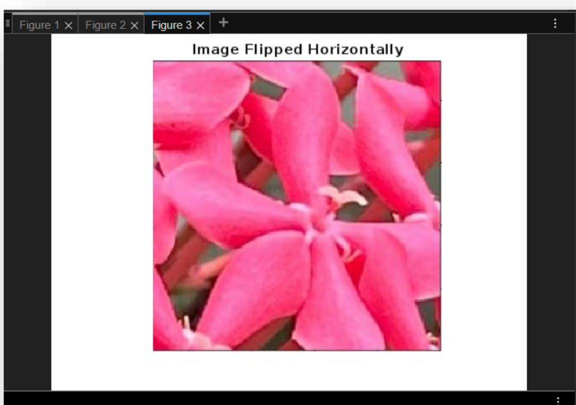

Figure 3. Image flipped horizontally in MATLAB

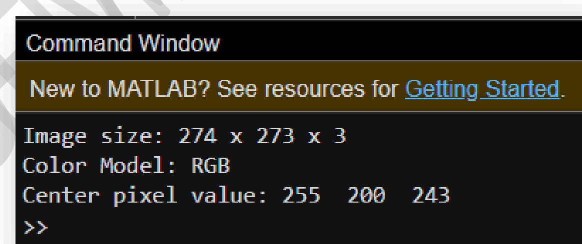

Figure 1. Image details in MATLAB

Octave Results

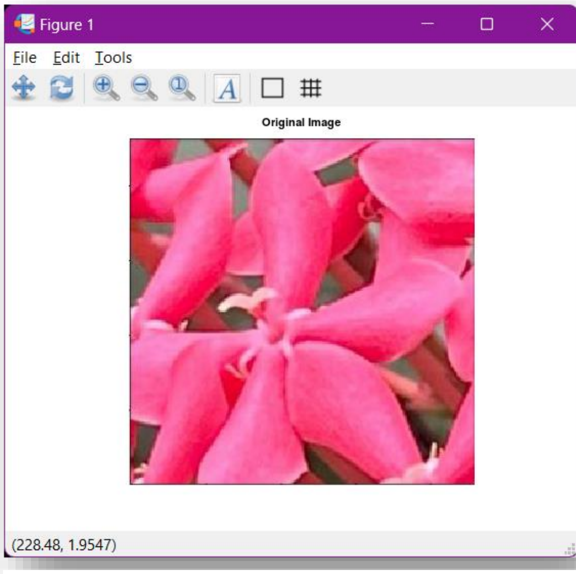

Figure 5. Acquiring an Image of a Flower in Octave

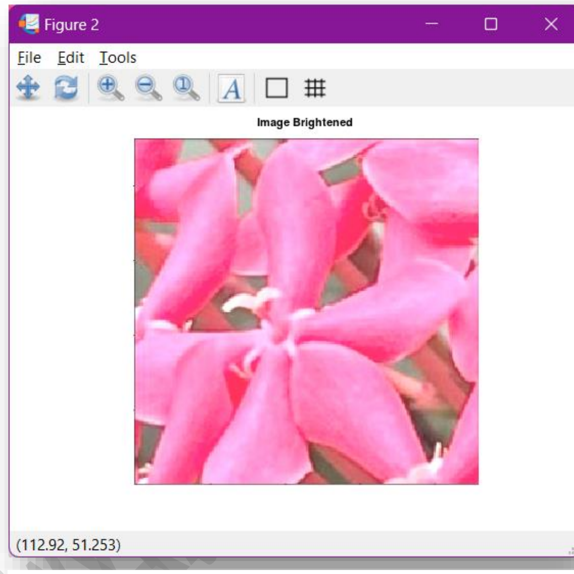

Figure 6. Image brightened in Octave

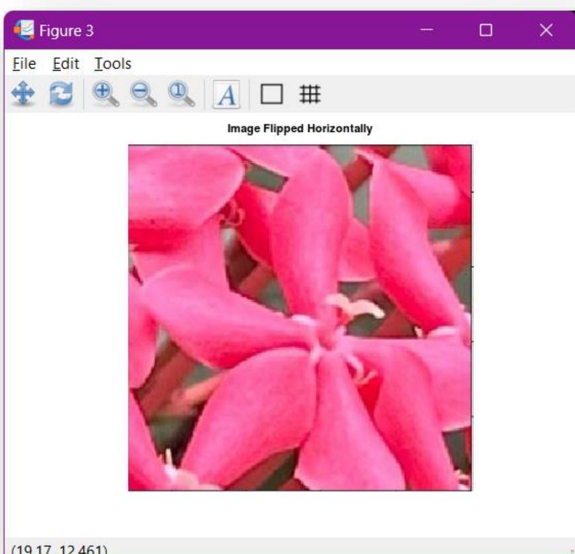

Figure 7. Image flipped horizontally in Octave

Command Window

```
>> Lab_Activity2_Octave

Image size: 274 x 273 x 3
Color Model: RGB
Center pixel value: 255 201 243
>>
```

Figure 8. Image details in Octave

OpenCV/Python Results

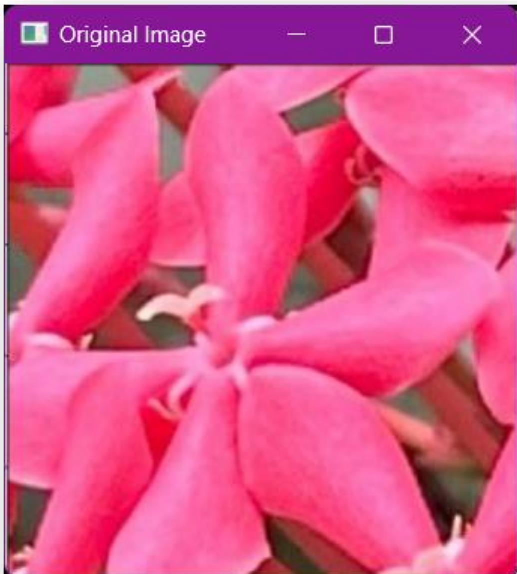

Figure 9. Acquiring an Image of a Flower in Python

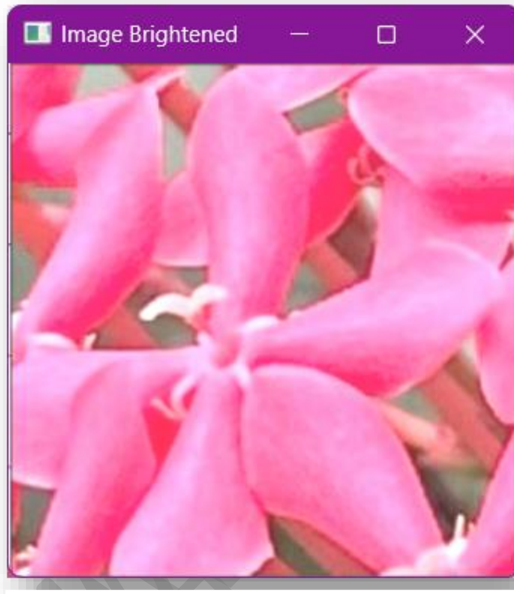

Figure 10. Image brightened in Python

Figure 11. Image flipped horizontally in Python

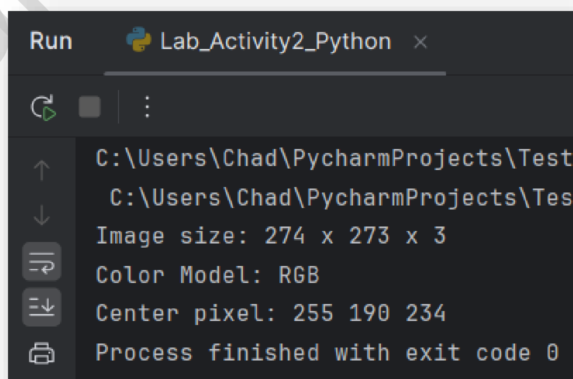

Figure 12. Image details in Python

These codes perform the following:

1. Reads an image using imread.
2. Displays the image using imshow.
3. Gets the image dimensions (rows, columns, color channels) using size and displays them.
4. Checks the color model (grayscale or RGB) based on the number of channels.
5. Accesses the value of a specific pixel (center pixel in this case).Performs a basic arithmetic operation (adding a constant value to all pixels) to brighten theimage.
6. Performs a basic geometric operation (flipping the image horizontally) using fliplr.

Parameter Modification

<You can modify it to explore other functionalities>

- Try displaying individual color channels for RGB images (e.g., imshow(img(:,:,1)) for red channel).
- Experiment with different arithmetic operations (subtraction, multiplication).
- Explore other geometric operations like image rotation (imrotate).

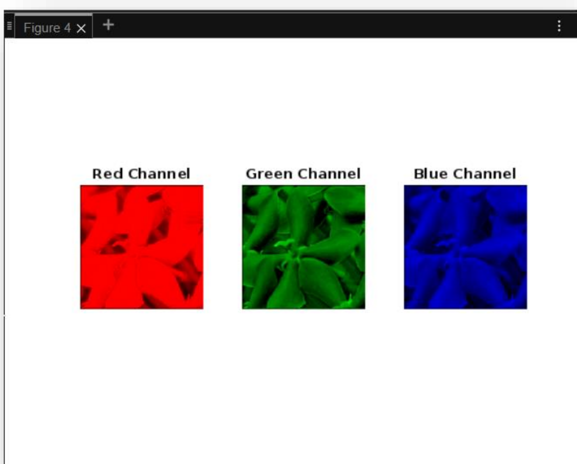

Figure 13. Image Channels in MATLAB

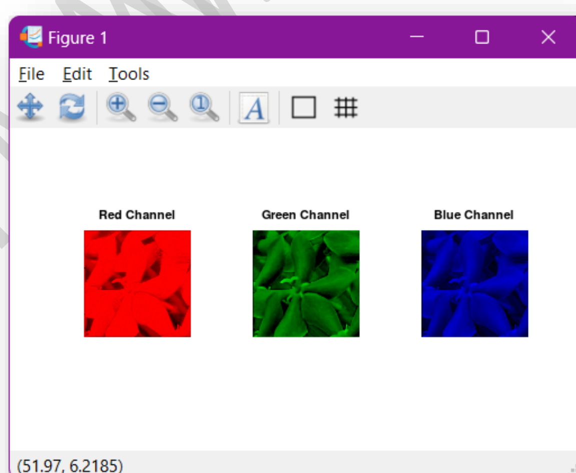

Figure 14. Image Channels in Octave

Figure 15. Image Channels in Python

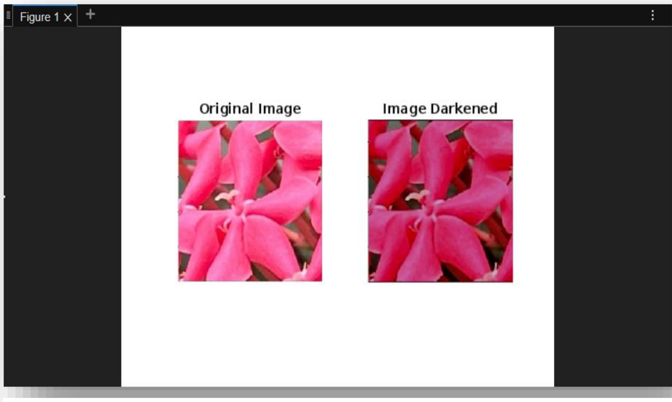

Figure 16. Image darkened using subtraction in MATLAB

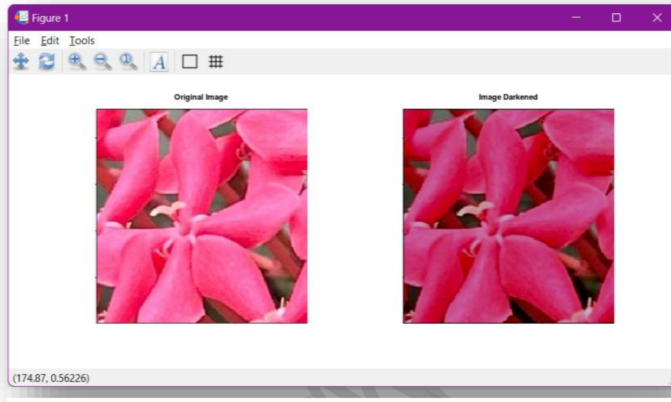

Figure 17. Image darkened using subtraction in Octave

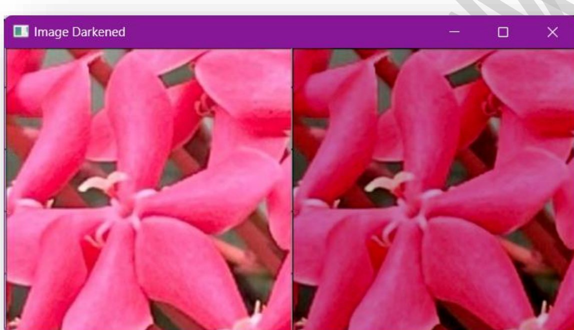

Figure 18. Image darkened using subtraction in Python

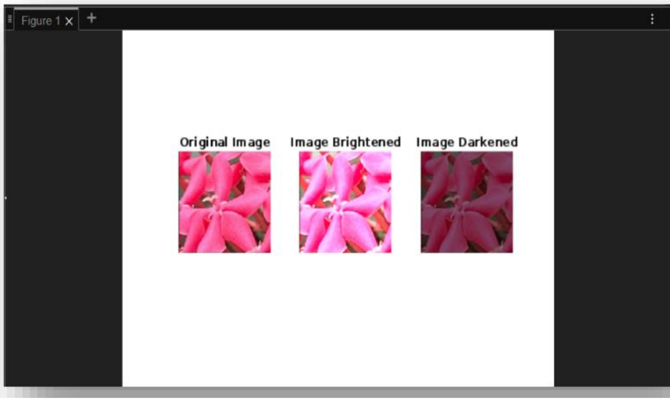

Figure 19. Image brightened and darkened using multiplication in MATLAB

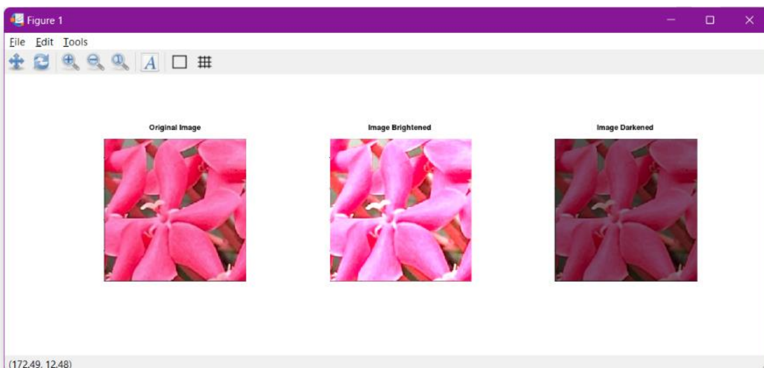

Figure 20. Image brightened and darkened using multiplication in Octave

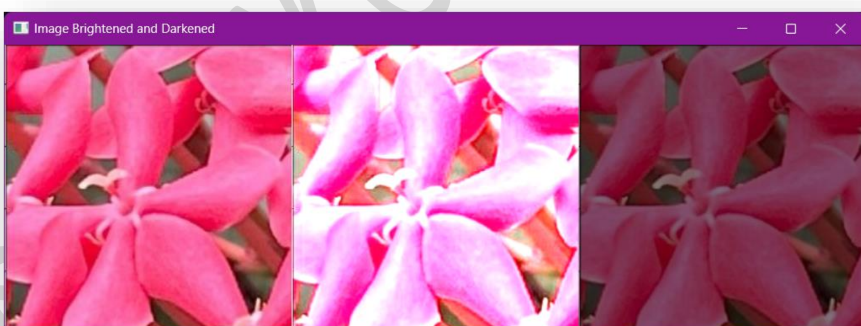

Figure 21. Image brightened and darkened using multiplication and division in Python

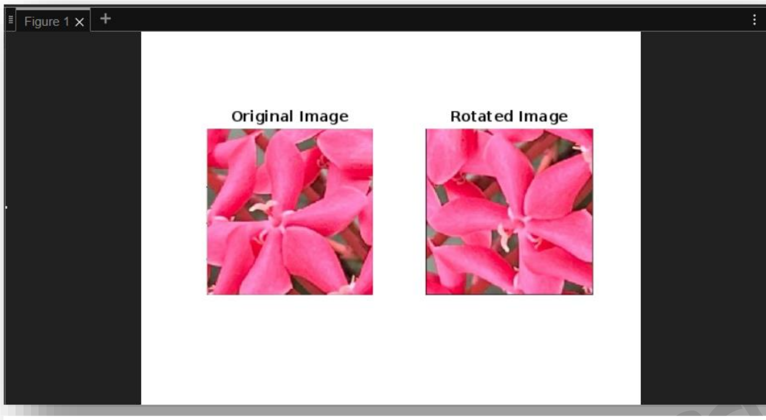

Figure 22. Image rotated by 90 degrees in MATLAB

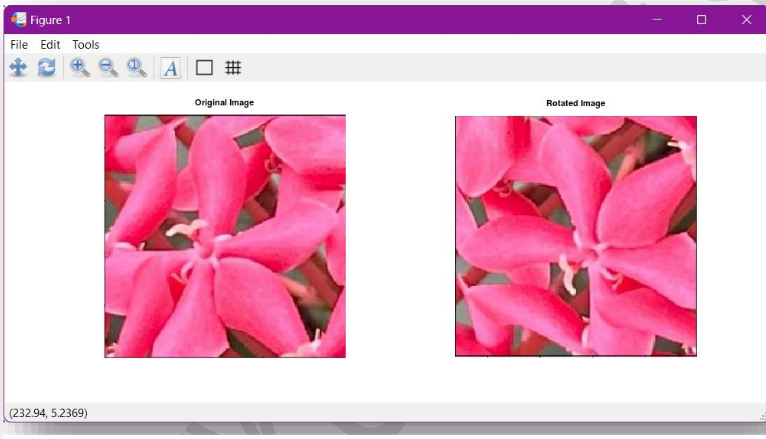

Figure 23. Image rotated by 90 degrees in Octave

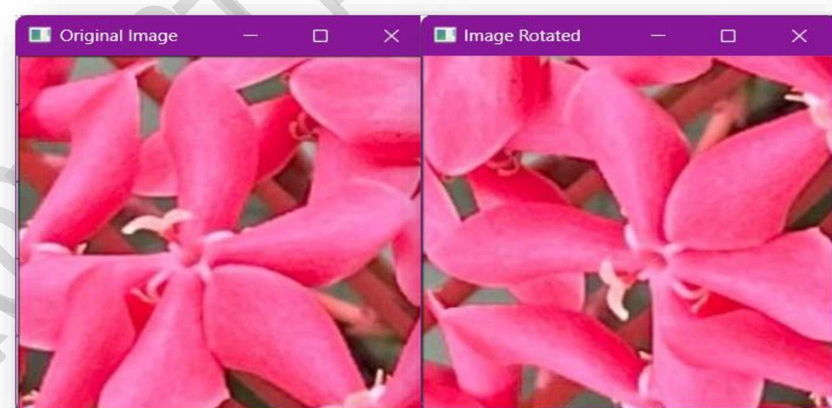

Figure 24. Image rotated by 90 degrees in Python

2. Visualize the results, analyze and interpret:

   < Discuss the effects of the applied algorithm on the image and its effectiveness in achieving the desired outcome. >

   In all of the platforms used, they were able to acquire the image perfectly. Additionally, they were also able to display the image, to obtain the dimensions of the image, and to check the color model of the image. All of the platforms used was also able to perform basic arithmetic operations like brightening the image and execute basic geometric operations like flipping the image horizontally. However, the values for the center pixel of the image vary depending on the platform used. Moreover, the discrepancy in displayed values for the center pixel of the image is miniscule between MATLAB and Octave. But, the shown values for the center pixel of the image in Python was different from the values for the center pixel displayed in MATLAB or Octave. We assume that the difference in results was due to how the image was read in OpenCV-Python. Additionally, we also tried to explore other functionalities. One of these functionalities is to display individual color channels for RGB images. As shown in the results above, all of the platforms were able to display the color channels of the image. Aside from displaying individual color channels for RGB images, we also tested other arithmetic operations such as subtraction and multiplication. Inverse to the addition which brightens the image, the subtraction darkens the image as shown in the results above. However, both addition and subtraction operations rely on the constant that will be added or subtracted in each element of the image's matrix to determine the contrast of the image. On the other hand, the multiplication operation sharpens or darkens the image depending on the constant. In multiplication operation, constant that is higher than 1 will sharpens the image while darkens the image when the constant is less than 1. Lastly, we also experimented with geometric operations like image rotation and all of the platforms were able to successfully rotate the image based on the desired outcome.

IV. Conclusion

Overall, this laboratory activity provided hands-on experience with essential image processing techniques and tools, laying a strong foundation for more advanced studies and applications in the field of image processing and computer vision. Throughout this activity, we achieved several key objectives such as acquiring the image and performing basic arithmetic and geometric operations which were vital in gaining deeper understanding in the field of image processing and computer vision. We also implement different algorithms to perform the objectives of the activity across different platforms which made us versatile when selecting the appropriate platform to perform image processing. The skills and knowledge gained from this exercise are invaluable for future projects and research involving digital images.

References

<This is in a separate page>

[1] D.J.D. Sayo. "University of the City of Manila Computer Engineering Department Honor Code," PLM-CpE Departmental Policies, 2020.

The AI University. (2020, June 18). *Make an Image Bright in OpenCV | Color Contrast in OpenCV* [Video]. https://www.youtube.com/watch?v=wDxmYKCNzE4

Coursesteach. (2023, November 25). *Computer Vision (Part 13)-Multiply image by a scaler and Blend 2 Images*. https://medium.com/@Coursesteach/computer-vision-part-13-multiply-by-a-scaler-60627d66c820