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

Elective 3

Laboratory Activity No. 1

Image Acquisition and Manipulation

Score

Submitted by:

Abarientos, Ramuel R.

Cazon, Iyhana Nicole A.

Reyes, John Alfred J.

Verzosa, Cristina Andrea B.

Vicente, Honesto E.

Saturday – 7:00 am – 4:00pm / CPE 0332.1-1

Date Submitted

27-07-2024

Submitted to:

Engr. Maria Rizette H. Sayo

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

I. Objectives

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

- Acquire the image.
- Rotate the image by 30 degrees.
- Flip the image horizontally.

II. Methods

A. Perform a task given in the presentation

- Copy and paste your MATLAB code

```
% Read the image img = imread('E:\PLM CET SUBJECTS\Digital Image
Processing\flower.jpg');
% Rotate by 45 degrees
rotated_img = imrotate(img, 45);

% Flip horizontally
flipped_img = fliplr(rotated_img);

% Display results
figure(1);
plot(1,1);
imshow(img);
title('Original Image');
figure(2);
plot(1,1);
imshow(rotated_img);
title('Rotated 45°'); figure(3); plot(1,1);
imshow(flipped_img); title('Rotated & Flipped');
```

MATLAB

New to MATLAB? See resources for Getting Started.

```
>> % Read the image
>> img = imread('C:\Users\user\Downloads\flower.jpg');
>> % Rotate by 30 degrees
>> rotated_img = imrotate(img, 30);
>>
>> % Flip horizontally
>> flipped_img = fliplr(rotated_img);
>>
>> % Display results
>> figure(1);
>> plot(1,1);
>> imshow(img);
>> title('Original Image');
>> figure(2);
>> plot(1,1);
>> imshow(rotated_img);
>> title('Rotated 30°');
>> figure(3);
>> plot(1,1);
>> imshow(flipped_img);
>> title('Rotated & Flipped');
>>
```

OCTAVE

Command Window

```
>> pkg load image;
>> img = imread('C:\Users\user\Downloads\flower.jpg');
>>
>> % Rotate by 30 degrees
>> rotated_img = imrotate(img, 30);
>>
>> % Flip horizontally
>> flipped_img = fliplr(rotated_img);
>>
>> % Display results
>> figure(1);
>> imshow(img);
>> title('Original Image');
>>
>> figure(2)
>> imshow(rotated_img);
>> title('Rotated 30 degrees');
>>
>> figure(3);
>> imshow(flipped_img);
>> title('Rotated & Flipped');
>> |
```

B. Supplementary Activity

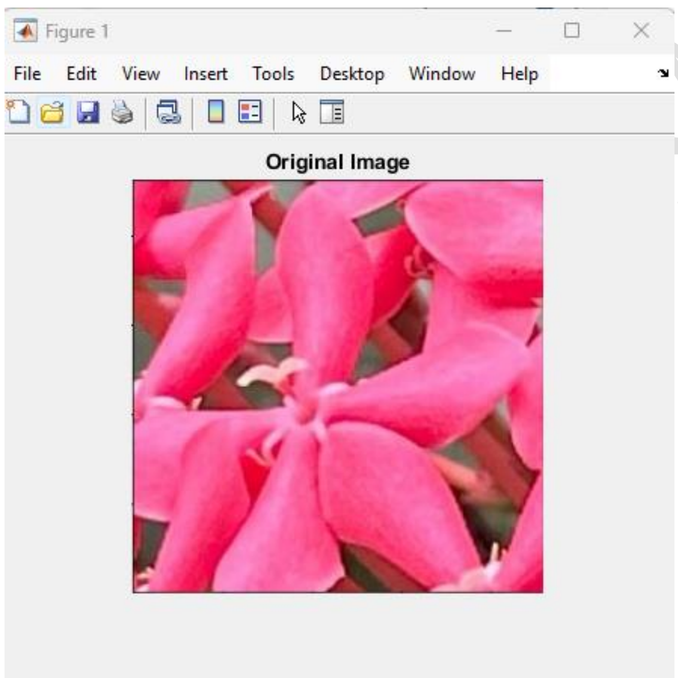
Python Code:

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

# Read the image
img = cv2.imread('E:/PLM CET SUBJECTS/Digital Image
Processing/flower.jpg')
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)  # Convert to RGB

# Rotate by 30 degrees
(h, w) = img.shape[:2]
center = (w // 2, h // 2)
M = cv2.getRotationMatrix2D(center, 30, 1.0)
rotated_img = cv2.warpAffine(img, M, (w, h))

# Flip horizontally
flipped_img = cv2.flip(rotated_img, 1)

# Display results
plt.figure(1)
plt.imshow(img)
plt.title('Original Image')

plt.figure(2)
plt.imshow(rotated_img)
plt.title('Rotated 30°')

plt.figure(3)
plt.imshow(flipped_img)
plt.title('Rotated & Flipped')

plt.show()
```

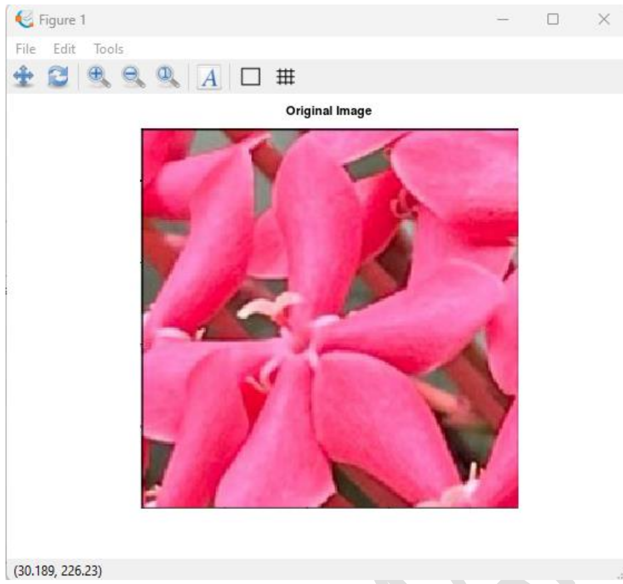
C. Results

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

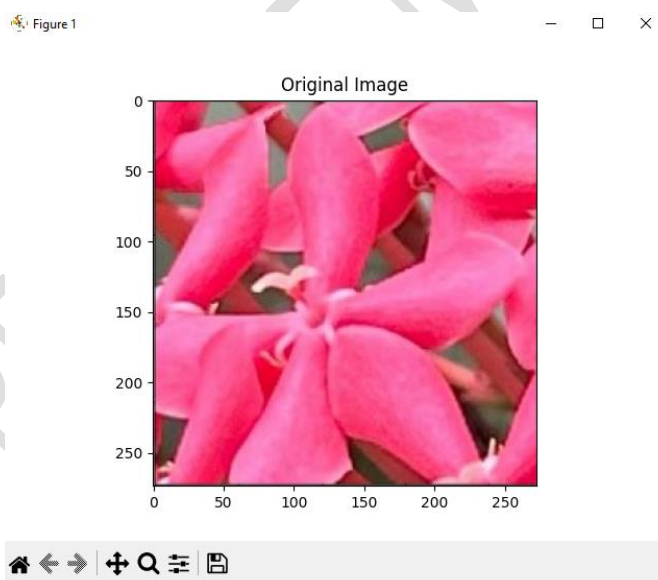
   picture file: flower.jpg



MATLAB

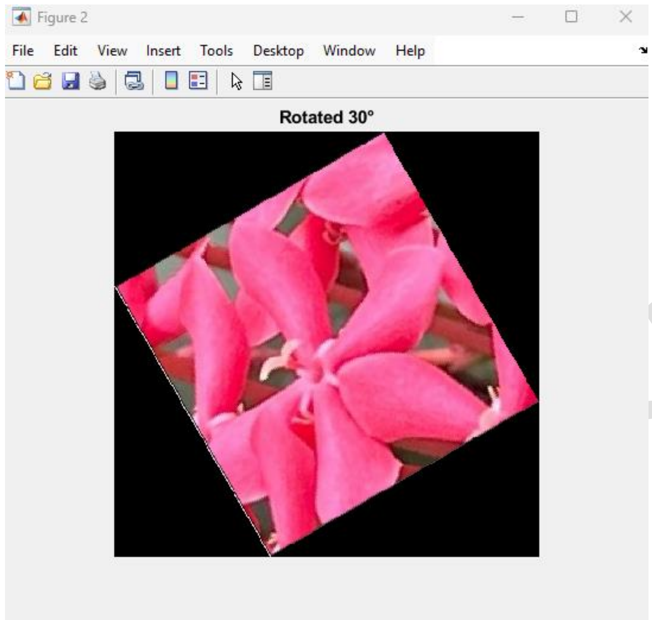


OCTAVE

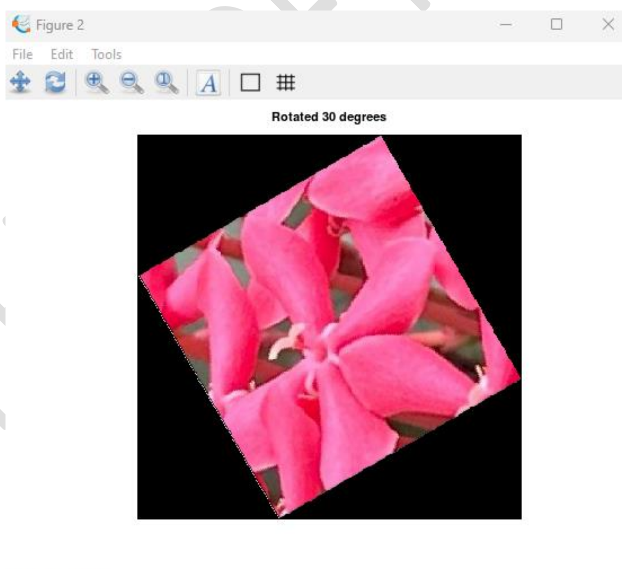


PYTHON

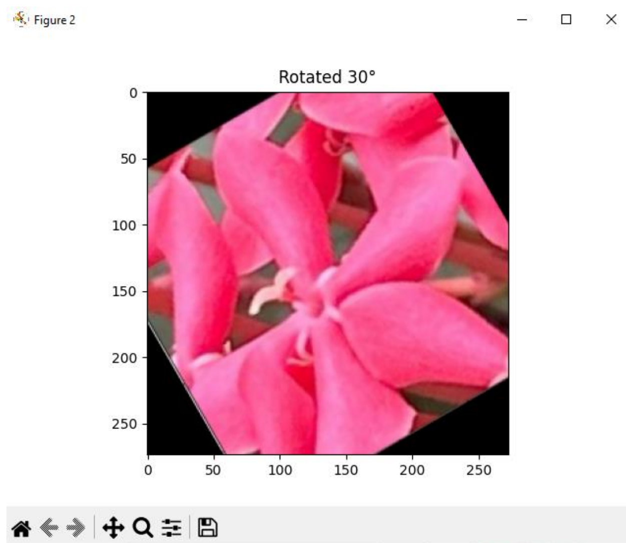
Figure 1: Acquire an Image of a Flower



MATLAB

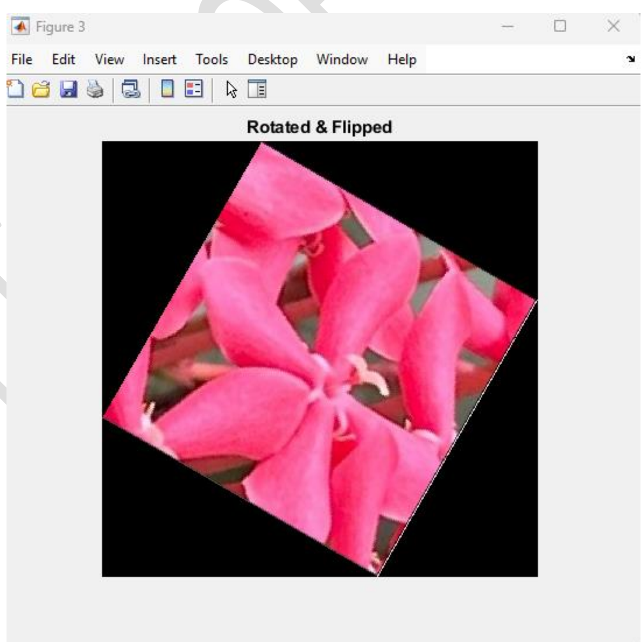


OCTAVE

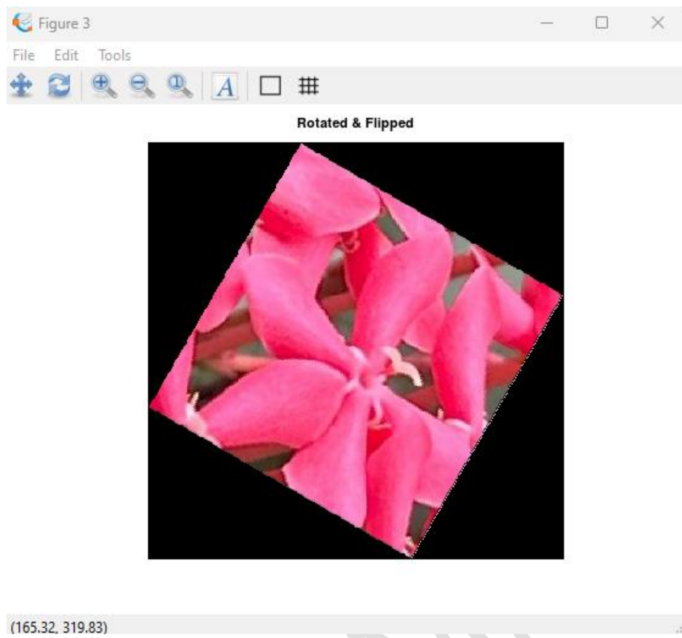


PYTHON

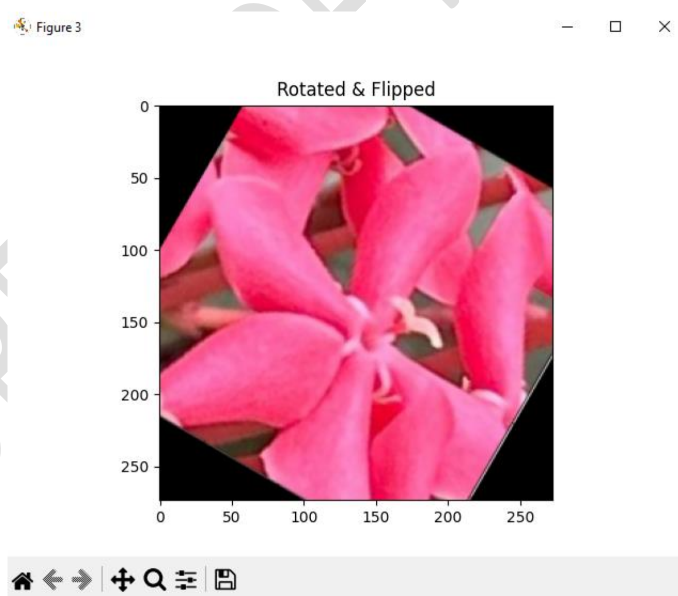
Figure 2: Rotate by 30 degrees



MATLAB



OCTAVE



PYTHON

Figure 3: Flip horizontally

2. Visualize the results, analyze and interpret:

The results of the image manipulation algorithms were visualized through a series of figures that demonstrated the original, rotated, and flipped images. The images clearly showed the changes applied, with the rotation effectively altering the orientation by 30 degrees and the horizontal flip inverting the image as expected.

Analyzing the rotated image, it was observed that the algorithm maintained the image quality without introducing significant distortions or artifacts. The rotation transformation was smooth, and the image's integrity was preserved, making the algorithm effective for applications requiring precise angle adjustments.

The horizontal flip effectively mirrored the image along the vertical axis, providing a clear demonstration of the transformation. This algorithm is particularly useful for symmetry analysis and correcting image orientations, proving its effectiveness in achieving the desired outcomes of the laboratory activity.

IV. Conclusion

In this laboratory activity, we successfully implemented the principles of image acquisition and manipulation using MATLAB/Octave and OpenCV with Python. We acquired an image and performed various transformations, including rotating the image by 30 degrees and flipping it horizontally. Each step was carried out using appropriate algorithms and functions available in the respective software environments.

Codes Used:

MATLAB:

% Rotate by 30 degrees

```
rotated_img = imrotate(img, 30);

% Flip horizontally

flipped_img = fliplr(rotated_img);
```

Octave:

```
% Rotate by 30 degrees

rotated_img = imrotate(img, 30);

% Flip horizontally

flipped_img = fliplr(rotated_img);
```

OpenCV with Python:

```
# Rotate by 30 degrees

(h, w) = img.shape[:2]

center = (w // 2, h // 2)

M = cv2.getRotationMatrix2D(center, 30, 1.0)

rotated_img = cv2.warpAffine(img, M, (w, h))

# Flip horizontally

flipped_img = cv2.flip(rotated_img, 1)
```

The results clearly demonstrated the effectiveness of these image processing techniques. The rotated images showed a precise 30-degree rotation, and the horizontally flipped images reflected the expected mirrored transformations. These operations are fundamental in image processing and are crucial for tasks such as image analysis, computer vision, and graphical manipulations.

Overall, the activity provided valuable hands-on experience with image processing tools and techniques, reinforcing theoretical concepts and highlighting their practical applications. The success confirmation of the tasks confirms the robustness of MATLAB/Octave and OpenCV in handling image manipulation tasks effectively.

References

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