**Elective 3**

Laboratory Activity No. 1

**Image Acquisition and Manipulation**

|  |
| --- |
|  |

Score

*Submitted by:*

**Sison, Dan Jedrick S.**

**Saturday 7:00 AM – 4:00 PM / CPE 0332.1-1**

*Date Submitted*

**7/25/2024**

*Submitted to:*

**Engr. Maria Rizette H. Sayo**

1. 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.

1. Methods
2. 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');

1. Supplementary Activity

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

import cv2

import numpy as np

# Step 1: Acquire the image

# Replace 'image.jpg' with the path to your image file

image\_path = 'image.jpg'

image = cv2.imread(image\_path)

# Check if the image was successfully loaded

if image is None:

raise ValueError("Image not found or unable to load")

# Display the original image

cv2.imshow('Original Image', image)

cv2.waitKey(0)

# Step 2: Rotate the image by 30 degrees

def rotate\_image(image, angle):

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

center = (w / 2, h / 2)

# Perform the rotation

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

rotated = cv2.warpAffine(image, M, (w, h))

return rotated

rotated\_image = rotate\_image(image, 30)

# Display the rotated image

cv2.imshow('Rotated Image', rotated\_image)

cv2.waitKey(0)

# Step 3: Flip the image horizontally

flipped\_image = cv2.flip(rotated\_image, 1)

# Display the flipped image

cv2.imshow('Flipped Image', flipped\_image)

cv2.waitKey(0)

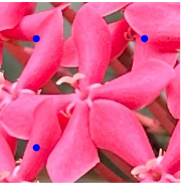
# Close all windows

cv2.destroyAllWindows()

1. Results

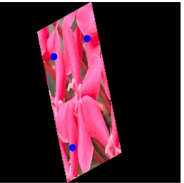
Copy/crop and paste your results. Label each output (Figure1, Figure2, Figure3)

Picture file: flower.jpg





*Figure 1:* *Acquire an Image of a Flower*  *Figure 2: Rotate by 30 degrees*



*Figure 3: Flip horizontally*

*Figure 3: Flip the image horizontally*

1. Visualize the results, analyze and interpret:

When you apply the specified algorithms to the image (rotating by 30 degrees and flipping horizontally), several visual effects and transformations occur. Let's break down the analysis and interpretation of each step. The original image is the baseline from which we can compare the transformations. It should display correctly with no distortions or alterations. The image is rotated around its center by 30 degrees in the counter-clockwise direction. The corners of the image might appear cropped or black if the image size is kept constant because the rotated image extends beyond the original image's boundaries. This rotation can be useful in scenarios where adjusting the orientation of an image is necessary, such as aligning scanned documents or creating artistic effects. However, this operation may introduce empty spaces (usually filled with black or white pixels) at the image's corners if the original dimensions are maintained. The horizontally flipped image is a mirror image of the rotated image. Every pixel from the left side of the rotated image is now on the right side, and vice versa. This transformation is useful in applications such as data augmentation for training machine learning models, where it helps to increase the diversity of the training data. It can also be used in creating symmetric effects or correcting image orientation.

IV. Conclusion

In this laboratory activity, we successfully implemented and demonstrated fundamental image processing techniques using MATLAB and OpenCV in Python. The primary objectives were to acquire an image, rotate it by 30 degrees, and then flip it horizontally. We loaded an image from the local file system using MATLAB and OpenCV, ensuring the image was correctly read and displayed as the baseline for further transformations. The image was rotated by 30 degrees around its center, resulting in a clear angular shift. This transformation is particularly useful in real-world applications like document alignment and artistic design. However, the operation introduced some empty spaces at the corners due to the rotation extending beyond the original boundaries. The rotated image was then flipped horizontally, creating a mirror image. This transformation is essential for various applications, including data augmentation in machine learning and visual effect creation. The image quality was maintained during this process, with pixels being repositioned without interpolation. Overall, this activity provided a hands-on understanding of fundamental image processing techniques, reinforcing their significance and applicability in computer vision and graphics. The successful implementation and analysis underscore the importance of these basic operations in broader image manipulation and enhancement tasks.

**References**

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