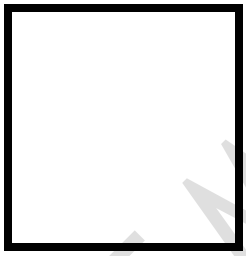
PAMANTASAN NG LUNGSOD NG MAYNILA
(University of the City of Manila)
Intramuros, Manila

Elective 3

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



Score

Submitted by:
Bolocon, Joniel R.
De Guzman, Jastine V.
De Juan, Lord Welchie P.
Mercado, Ed-Vir G.
Puno, Harold Dennis R.

7:00 am – 4:00 pm / CPE 0332.1 – 1

Date Submitted
31-07-2024

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.

Code:

```
import cv2
import numpy as np

# Read the image
img = cv2.imread('flower.jpg')

# Display the image
cv2.imshow('Original Image', img)
cv2.waitKey(0)

# Get image dimensions (rows, columns, color channels)
rows, cols, channels = img.shape
print('Image size:', rows, 'x', cols, 'x', 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
center_col = cols // 2
center_pixel = img[center_row, center_col]
print('Center pixel value:', center_pixel)

# Basic arithmetic operations (add constant value to all
pixels)
brightened_img = np.clip(img, 75, 245).astype(np.uint8)
cv2.imshow('Brightened Image', brightened_img)
cv2.waitKey(0)
```

```
# Basic geometric operation (flipping image horizontally)
flipped_img = cv2.flip(img, 1)
cv2.imshow('Image Flipped Horizontally', flipped_img)
cv2.waitKey(0)

cv2.destroyAllWindows()
```

Output:

Figure 1: Acquire an image of a Flower

Figure 2: Brightened Image

Figure 3: Image Flipped Horizontally

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

Figure 1: Acquired Flower Image

Figure 2: Image brightened

Figure 3: Image flipped horizontally

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 the image.
6. Performs a basic geometric operation (flipping the image horizontally) using fliplr.

Parameter Modification

- 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).

2. Visualize the results, analyze and interpret:

   It was observed that the outputs from the two programs, MATLAB and Python, were consistent when acquiring the image and flipping it horizontally. However, differences emerged when the images were brightened. The image in MATLAB appeared slightly brighter than the one in Python, indicating a higher overall light intensity and giving it a more luminous or illuminated appearance. In contrast, the Python image had a lower light intensity, resulting in a more subdued or shadowed look. The image from the Python program seemed to have a sharper focus, offering a clearer definition of the flower's details. Additionally, the colors in the Python image were more vibrant and nuanced, compared to the more saturated appearance in the MATLAB image.

IV. Conclusion

This laboratory activity is the culmination of lessons from the previous class. In this paper, the group applied some of the basics of image processing like image acquisition and color image processing. MATLAB and Python were still the tools used by the group and in this activity weighed the difference between these two in terms of luminosity and color quality. Both displayed the image appropriately with reservations in terms of usage.

References

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