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

Elective 3

Laboratory Activity No. 3
Image Enhancement

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
2-08-2024

Submitted to:

Engr. Maria Rizette H. Sayo

I. Objectives

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

1. Acquire the image.
2. Show histogram equalization.
3. Show contrast enhancement.
4. Show filtering in the spatial domain (average and median).

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 original image figure;
imshow(img); title('Original
Image');

% Convert to grayscale if the image is RGB if
size(img, 3) == 3
    img_gray = rgb2gray(img);
else
    img_gray = img;
end

% Display the grayscale image
figure;
imshow(img_gray);
title('Grayscale Image');

% Contrast enhancement using imadjust
img_contrast_enhanced = imadjust(img_gray);

% Display the contrast-enhanced image
figure; imshow(img_contrast_enhanced);
title('Contrast Enhanced Image (imadjust)');

% Histogram equalization img_histeq
= histeq(img_gray);

% Display the histogram equalized image
figure;
```

```
imshow(img_histeq);
title('Equalized Image');

% Filtering using average filterh
h_avg = fspecial('average', [5, 5]);
img_avg_filtered = imfilter(img_gray, h_avg);

% Display the average filtered image
figure;
imshow(img_avg_filtered);
title('Filtered Image (Average)');

% Filtering using median filter img_median_filtered =
medfilt2(img_gray, [5, 5]);

% Display the median filtered image
figure; imshow(img_median_filtered);
title('Filtered Image (Median)');

% Display histograms for comparison

% Grayscale histogram
figure;
imhist(img_gray);
title('Histogram of Grayscale');

% Enhanced histogram (imadjust) figure;
imhist(img_contrast_enhanced);
title('Histogram of Enhanced Image');

% Equalized histogram
figure;
imhist(img_histeq);
title('Histogram of Equalized Image');

% Histogram (Average Filtered)
figure; imhist(img_avg_filtered);
title('Histogram of Average Filtered)');

% Histogram (Median Filtered)
figure;
imhist(img_median_filtered);
title('Histogram of Median Filtered)');
```

PAMANTASAN NG LUNGSOD NG MAYNILA

(University of the City of Manila)

Intramuros, Manila

Editor - C:\Users\user\labact3_matlab.m

LabAct2.m | image_color.m | image_color1.m | lab3.m | lab_act_3_mod

```
% Read an image
img = imread('C:\Users\user\Downloads\LabAct3\flower.jpg');

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

% Convert to grayscale if the image is RGB
if size(img, 3) == 3
    img_gray = rgb2gray(img);
else
    img_gray = img;
end

% Display the grayscale image
figure;
imshow(img_gray);
title('Grayscale Image');

% Contrast enhancement using imadjust
img_contrast_enhanced = imadjust(img_gray);

% Display the contrast-enhanced image
figure;
imshow(img_contrast_enhanced);
title('Contrast Enhanced Image (imadjust)');

% Histogram equalization
img_histeq = histeq(img_gray);
```

LabAct2.m | image_color.m | image_color1.m | lab3.m | lab_a

```
% Histogram equalization
img_histeq = histeq(img_gray);

% Display the histogram equalized image
figure;
imshow(img_histeq);
title('Equalized Image');

% Filtering using average filter
h_avg = fspecial('average', [5, 5]);
img_avg_filtered = imfilter(img_gray, h_avg);

% Display the average filtered image
figure;
imshow(img_avg_filtered);
title('Filtered Image (Average)');

% Filtering using median filter
img_median_filtered = medfilt2(img_gray, [5, 5]);

% Display the median filtered image
figure;
imshow(img_median_filtered);
title('Filtered Image (Median)');

% Display histograms for comparison

% Grayscale histogram
figure;
imhist(img_gray);
```

Editor - C:\Users\user\labact3_matlab.m

LabAct2.m | image_color.m | image_color1.m | lab3.m | lab_act_

```
% Display histograms for comparison

% Grayscale histogram
figure;
imhist(img_gray);
title('Histogram of Grayscale Image');

% Enhanced histogram (imadjust)
figure;
imhist(img_contrast_enhanced);
title('Histogram of Enhanced Image (imadjust)');

% Equalized histogram
figure;
imhist(img_histeq);
title('Histogram of Equalized Image');

% Histogram (Average Filtered)
figure;
imhist(img_avg_filtered);
title('Histogram of Average Filtered Image');

% Histogram (Median Filtered)
figure;
imhist(img_median_filtered);
title('Histogram of Median Filtered Image');
```

Matlab

lab_act_3_modif.m | labact3_octave.m

```
pkg load image

% Read an image
img = imread('C:\Users\user\Downloads\LabAct3\flower.jpg');

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

% Convert to grayscale if the image is RGB
if size(img, 3) == 3
    img_gray = rgb2gray(img);
else
    img_gray = img;
end

% Display the grayscale image
figure;
imshow(img_gray);
title('Grayscale Image');

% Contrast enhancement using imadjust
img_contrast_enhanced = imadjust(img_gray);

% Display the contrast-enhanced image
figure;
imshow(img_contrast_enhanced);
title('Contrast Enhanced Image (imadjust)');

% Histogram equalization
img_histeq = histeq(img_gray);

% Display the histogram equalized image
figure;
imshow(img_histeq);
title('Equalized Image');

% Filtering using average filter
h_avg = fspecial('average', [5, 5]);
img_avg_filtered = imfilter(img_gray, h_avg);

% Display the average filtered image
figure;
imshow(img_avg_filtered);
title('Filtered Image (Average)');

% Filtering using median filter
img_median_filtered = medfilt2(img_gray, [5, 5]);
```

lab_act_3_modif.m | labact3_octave.m

```
imshow(img_median_filtered);
title('Filtered Image (Median)');

% Display histograms for comparison

% Grayscale histogram
figure;
imhist(img_gray);
title('Histogram of Grayscale Image');

% Enhanced histogram (imadjust)
figure;
imhist(img_contrast_enhanced);
title('Histogram of Enhanced Image (imadjust)');

% Equalized histogram
figure;
imhist(img_histeq);
title('Histogram of Equalized Image');

% Histogram (Average Filtered)
figure;
imhist(img_avg_filtered);
title('Histogram of Average Filtered Image');

% Histogram (Median Filtered)
figure;
imhist(img_median_filtered);
title('Histogram of Median Filtered Image');
```

Octave

B. Supplementary Activity

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

```
import cv2
import numpy as np
import matplotlib.pyplot as plt
from skimage import exposure

# Read an image
img = cv2.imread('E:/PLM CET SUBJECTS/Digital Image Processing/flower.jpg')

# Display the original image
plt.figure()
plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.title('Original Image')
```

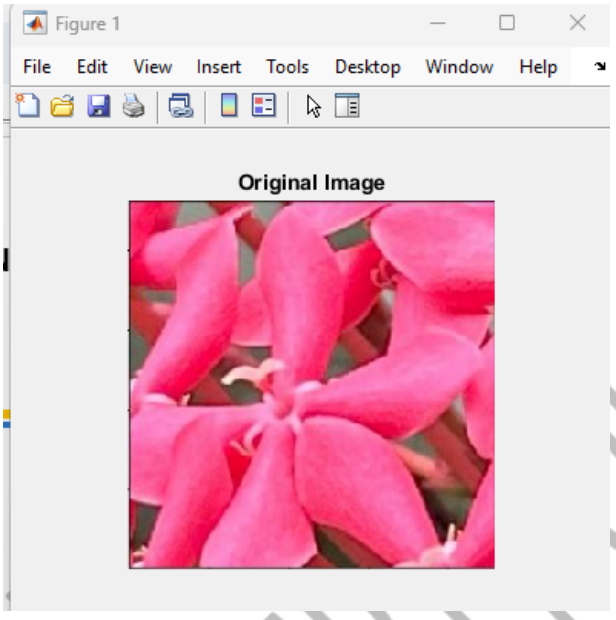
```
plt.show()

# Convert to grayscale if the image is RGB
if len(img.shape) == 3:
    img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
else:
    img_gray = img

# Display the grayscale image
plt.figure()
plt.imshow(img_gray, cmap='gray')
plt.title('Grayscale Image')
plt.show()

# Contrast enhancement using imadjust equivalent in Python
img_contrast_enhanced = exposure.rescale_intensity(img_gray,
in_range=(img_gray.min(), img_gray.max()))

# Display the contrast-enhanced image
plt.figure()
plt.imshow(img_contrast_enhanced, cmap='gray')
plt.title('Contrast Enhanced Image (imadjust)')
plt.show()

# Histogram equalization
img_histeq = cv2.equalizeHist(img_gray)

# Display the histogram equalized image
plt.figure()
plt.imshow(img_histeq, cmap='gray')
plt.title('Equalized Image')
plt.show()

# Filtering using average filter
h_avg = np.ones((5, 5), np.float32) / 25
img_avg_filtered = cv2.filter2D(img_gray, -1, h_avg)

# Display the average filtered image
plt.figure()
plt.imshow(img_avg_filtered, cmap='gray')
plt.title('Filtered Image (Average)')
plt.show()

# Filtering using median filter
img_median_filtered = cv2.medianBlur(img_gray, 5)

# Display the median filtered image
plt.figure()
plt.imshow(img_median_filtered, cmap='gray')
plt.title('Filtered Image (Median)')
```

```
plt.show()

# Display histograms for comparison
fig, axes = plt.subplots(3, 2, figsize=(12, 8))

axes[0, 0].hist(img_gray.ravel(), bins=256, histtype='step', color='black')
axes[0, 0].set_title('Histogram of Grayscale')

axes[0, 1].imshow(img_gray, cmap='gray')
axes[0, 1].set_title('Grayscale Image')

axes[1, 0].hist(img_contrast_enhanced.ravel(), bins=256, histtype='step',
color='black')
axes[1, 0].set_title('Histogram of Enhanced Image')

axes[1, 1].imshow(img_contrast_enhanced, cmap='gray')
axes[1, 1].set_title('Contrast Enhanced Image')

axes[2, 0].hist(img_histeq.ravel(), bins=256, histtype='step', color='black')
axes[2, 0].set_title('Histogram of Equalized Image')

axes[2, 1].imshow(img_histeq, cmap='gray')
axes[2, 1].set_title('Equalized Image')

fig.tight_layout()
plt.show()

# Display histograms for filtered images
fig, axes = plt.subplots(2, 2, figsize=(12, 8))

axes[0, 0].hist(img_avg_filtered.ravel(), bins=256, histtype='step',
color='black')
axes[0, 0].set_title('Histogram of Average Filtered')

axes[0, 1].imshow(img_avg_filtered, cmap='gray')
axes[0, 1].set_title('Average Filtered Image')

axes[1, 0].hist(img_median_filtered.ravel(), bins=256, histtype='step',
color='black')
axes[1, 0].set_title('Histogram of Median Filtered')

axes[1, 1].imshow(img_median_filtered, cmap='gray')
axes[1, 1].set_title('Median Filtered Image')

fig.tight_layout()
plt.show()
```

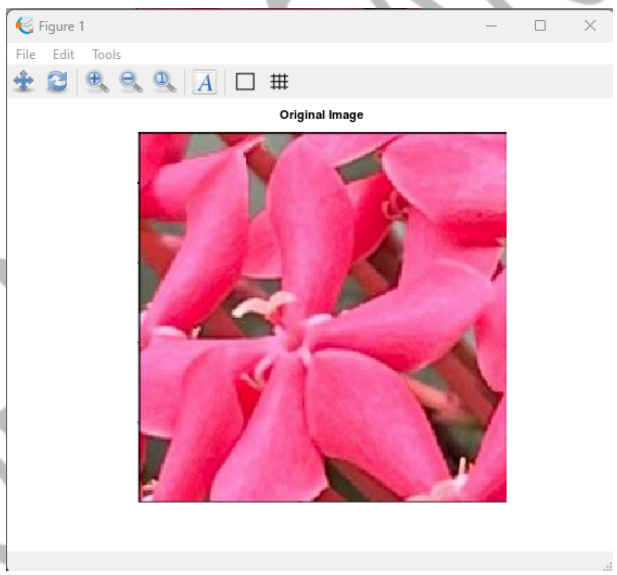
III. Results

Steps:

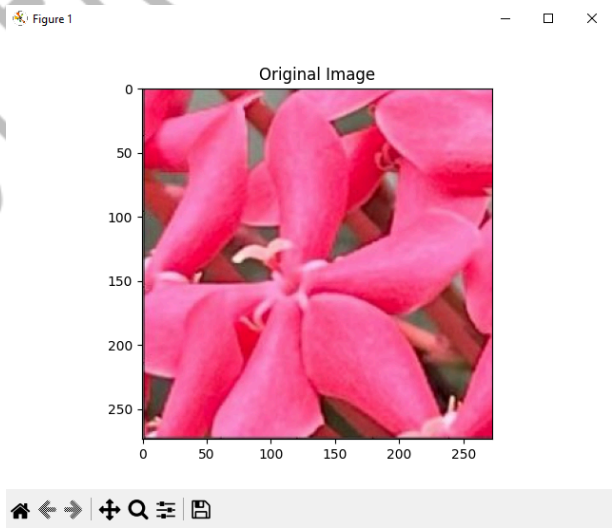
1. Copy/crop and paste your results. Label each output (Figure1, Figure2, Figure3, Figure 4, and Figure 5)



Matlab



Octave



Python

Figure 1: Acquire an Image of a Flower

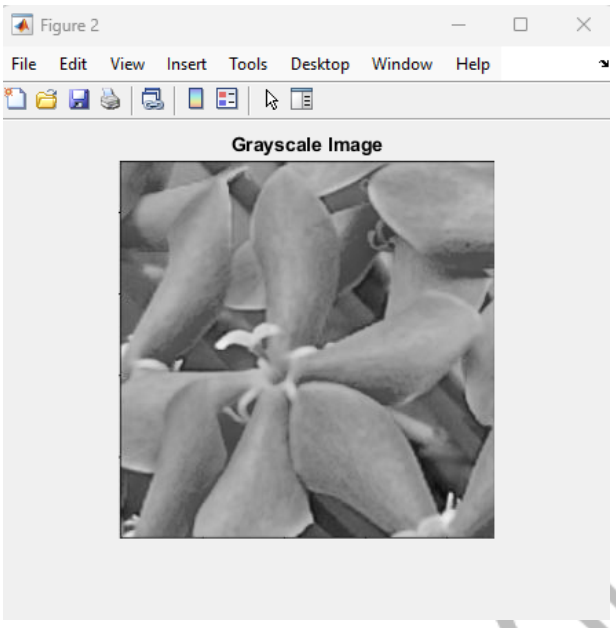


Figure 2
File Edit View Insert Tools Desktop Window Help
Grayscale Image

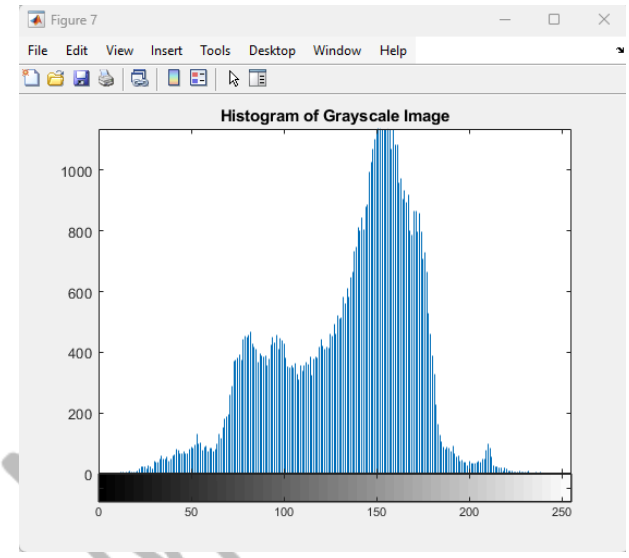


Figure 7
File Edit View Insert Tools Desktop Window Help
Histogram of Grayscale Image
1000
800
600
400
200
0
0
50
100
150
200
250

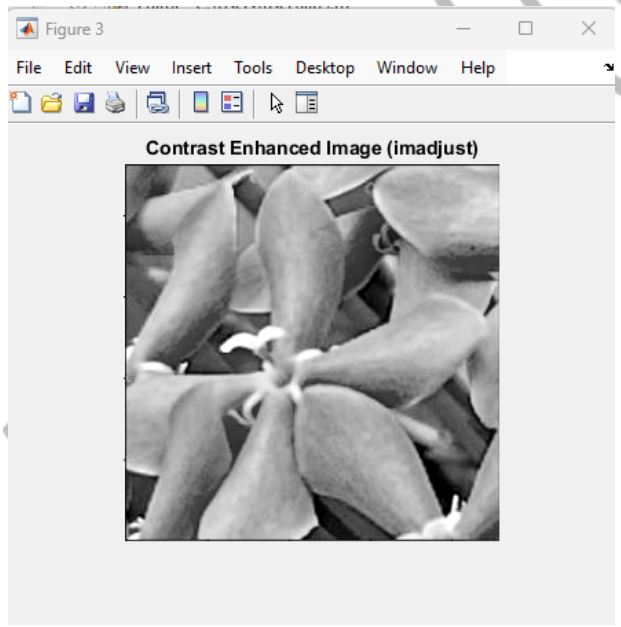


Figure 3
File Edit View Insert Tools Desktop Window Help
Contrast Enhanced Image (imadjust)

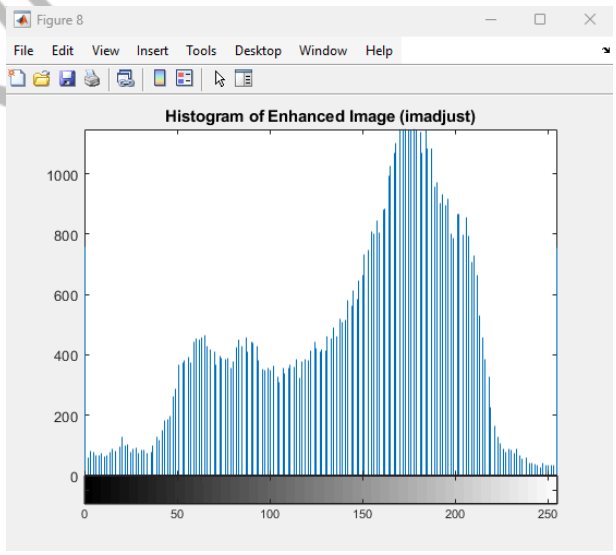


Figure 8
File Edit View Insert Tools Desktop Window Help
Histogram of Enhanced Image (imadjust)
1000
800
600
400
200
0
0
50
100
150
200
250

Matlab

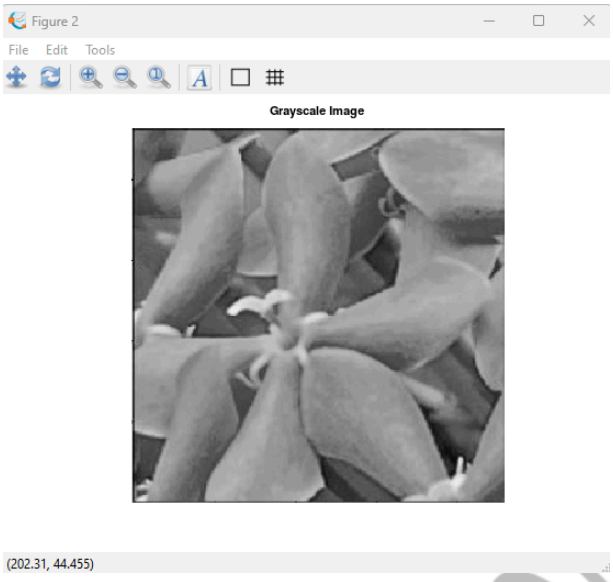


Figure 2
File Edit Tools
Grayscale Image
(202.31, 44.455)

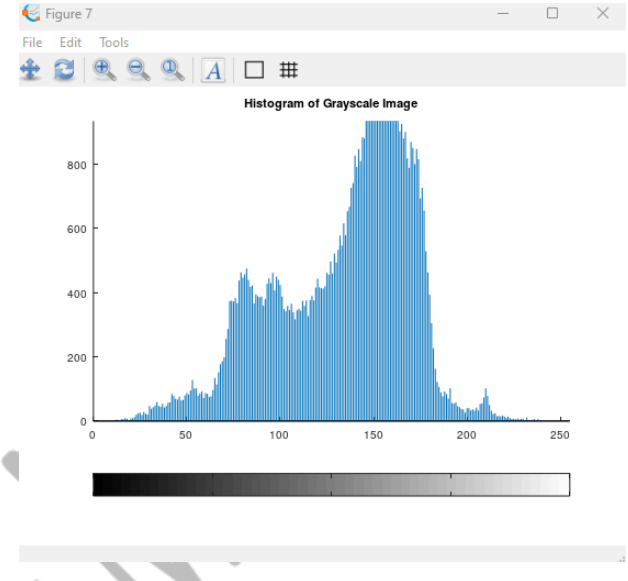


Figure 7
File Edit Tools
Histogram of Grayscale Image
800
600
400
200
0
0
50
100
150
200
250

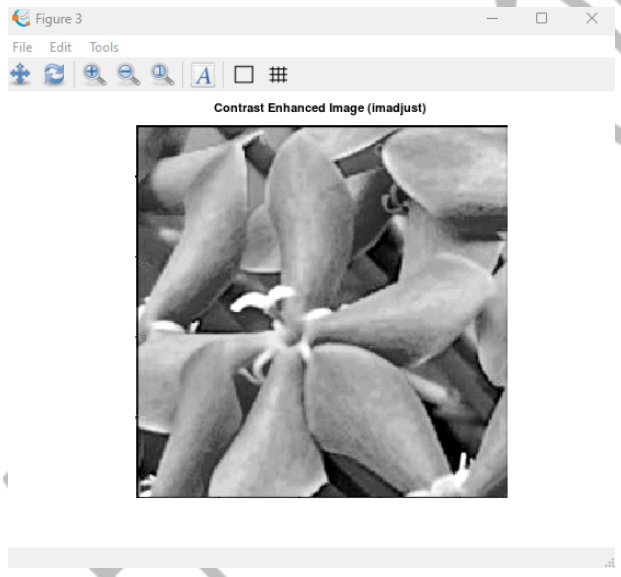


Figure 3
File Edit Tools
Contrast Enhanced Image (imadjust)

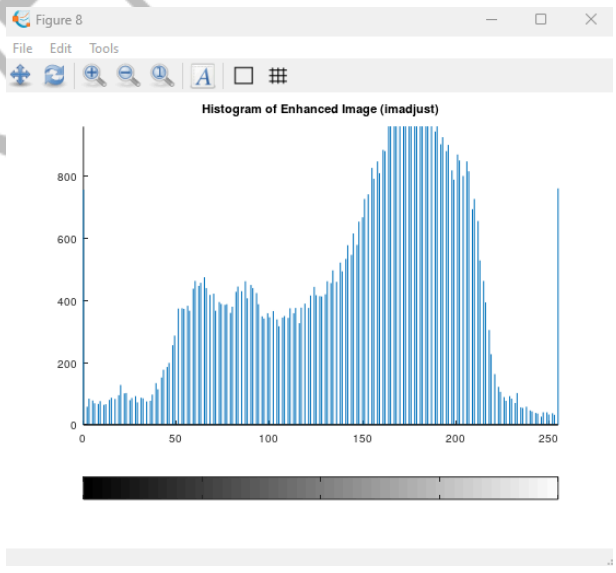
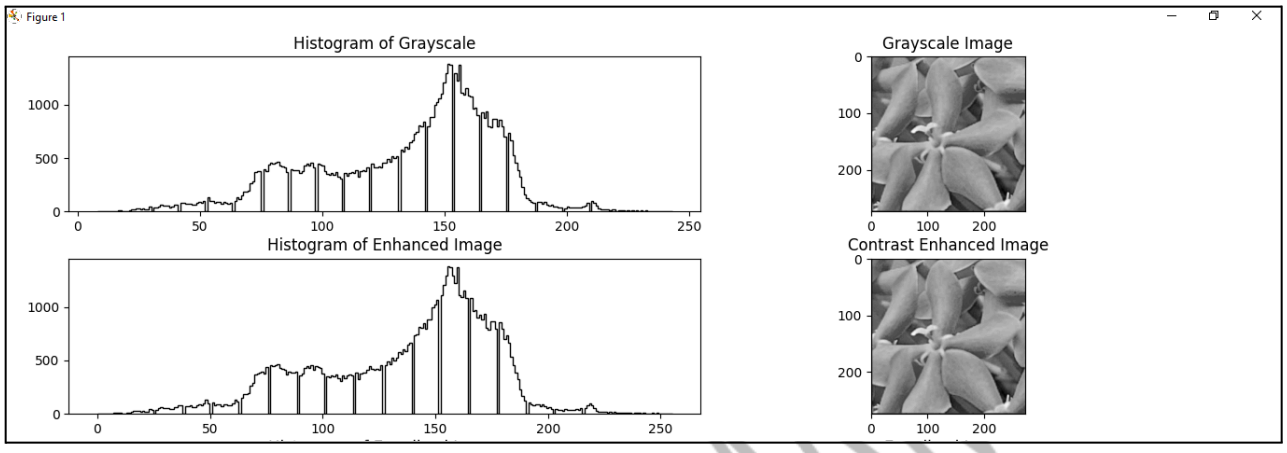


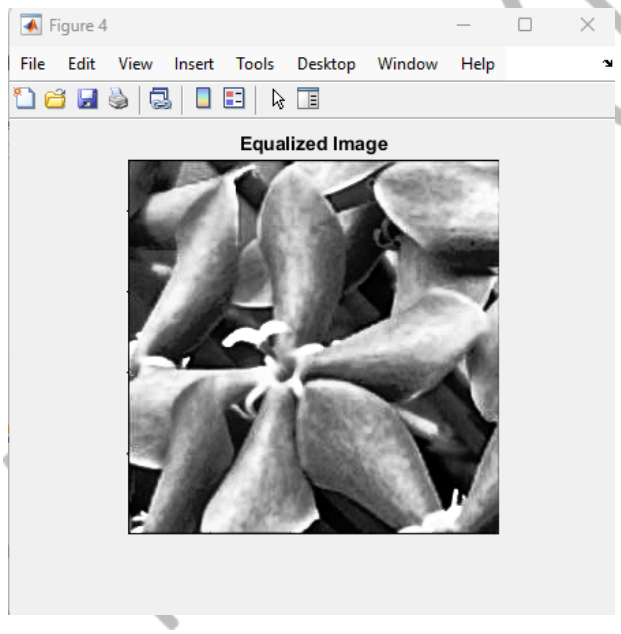
Figure 8
File Edit Tools
Histogram of Enhanced Image (imadjust)
800
600
400
200
0
0
50
100
150
200
250

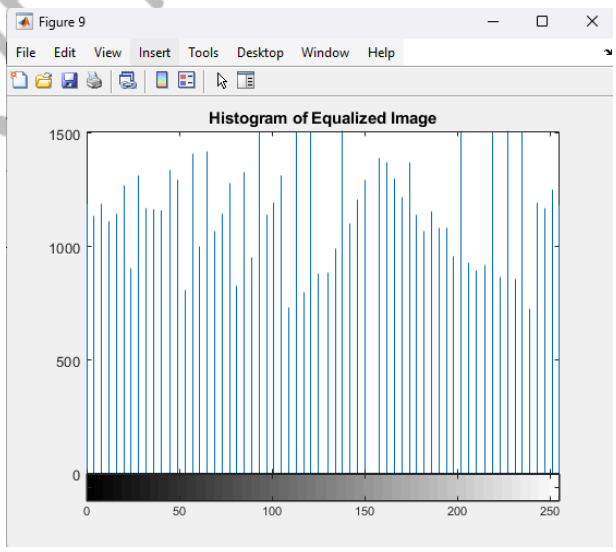
Octave



Python

Figure 2: Grayscale, Contrast Enhancement, and its Histogram





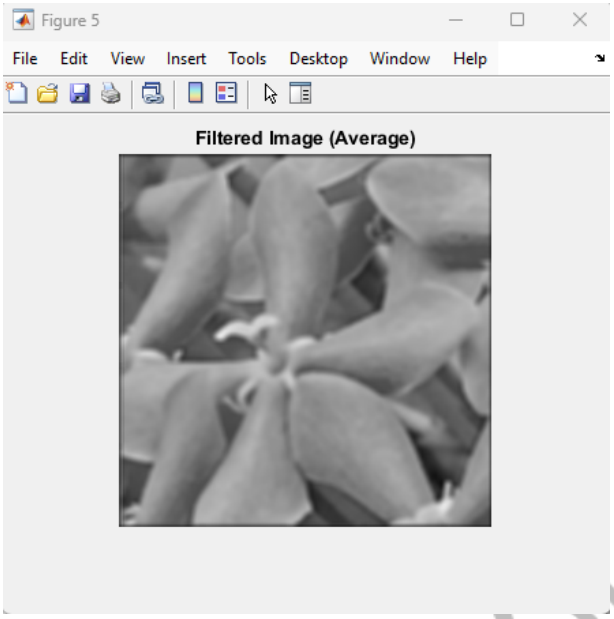


Figure 5
File Edit View Insert Tools Desktop Window Help
Filtered Image (Average)

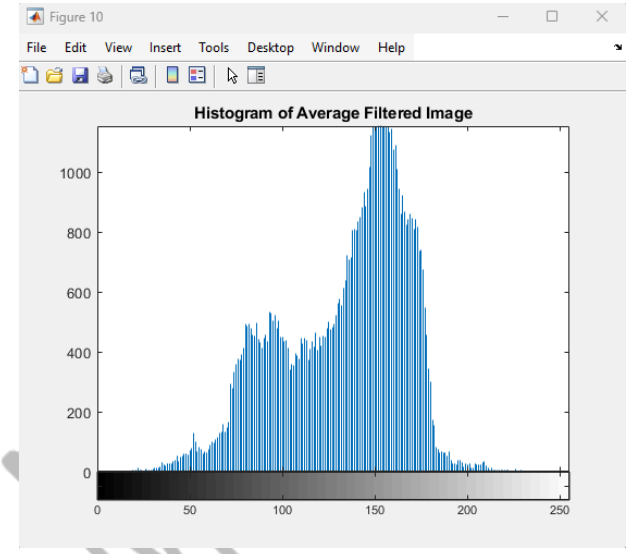


Figure 10
File Edit View Insert Tools Desktop Window Help
Histogram of Average Filtered Image
1000
800
600
400
200
0
0
50
100
150
200
250

Matlab

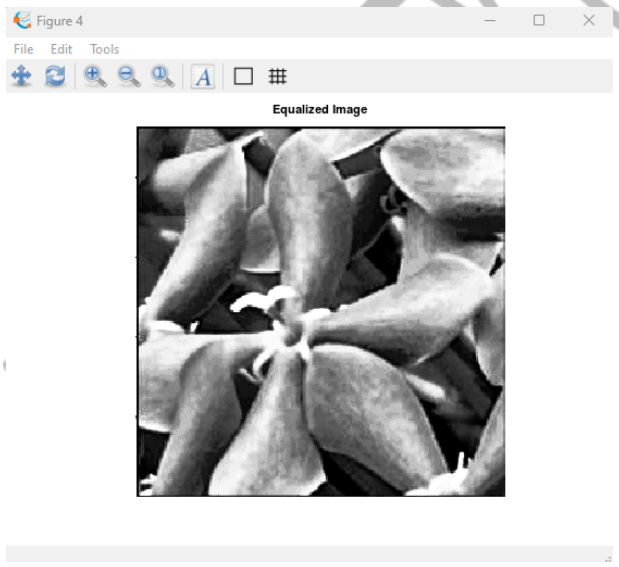


Figure 4
File Edit Tools
Equalized Image

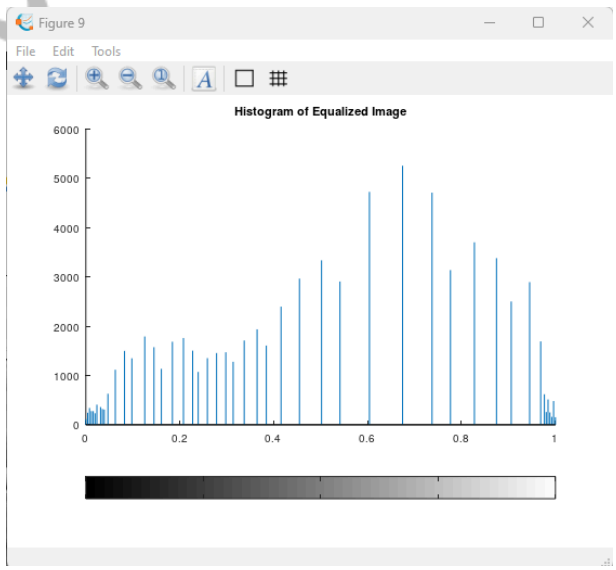
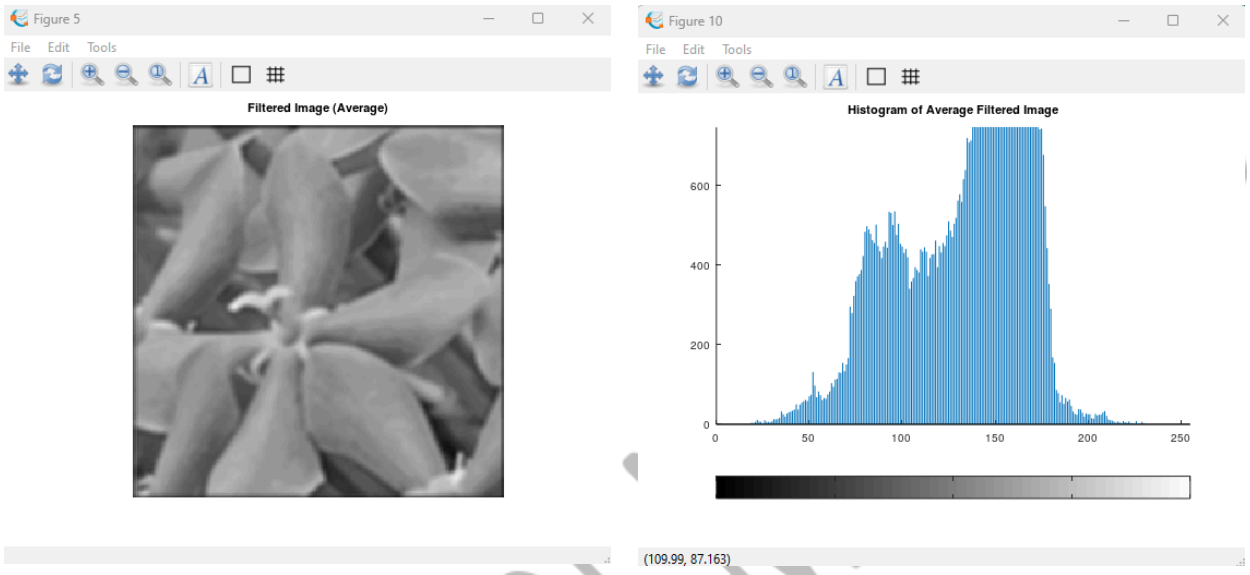
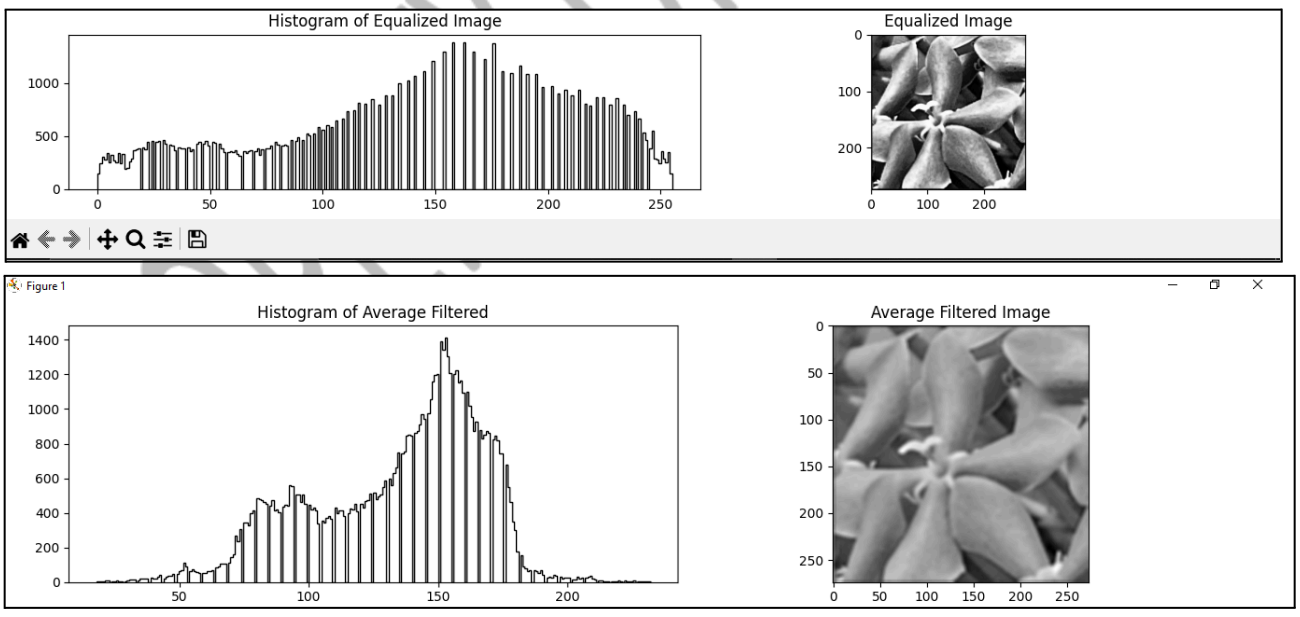


Figure 9
File Edit Tools
Histogram of Equalized Image
6000
5000
4000
3000
2000
1000
0
0
0.2
0.4
0.6
0.8
1

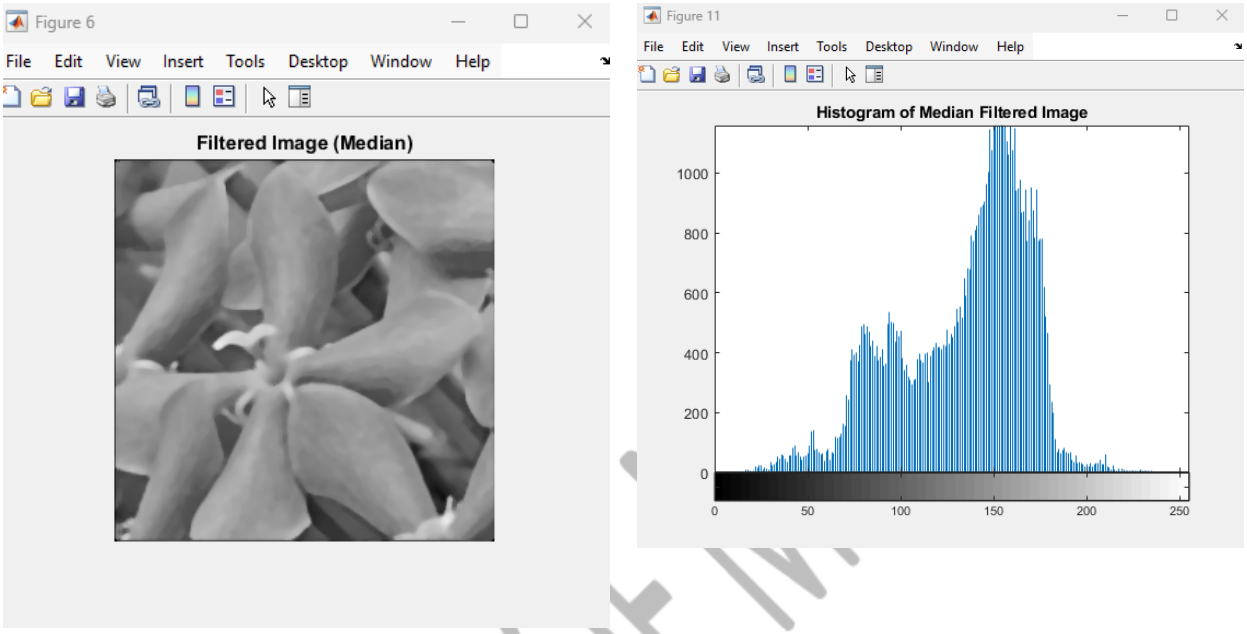


Octave

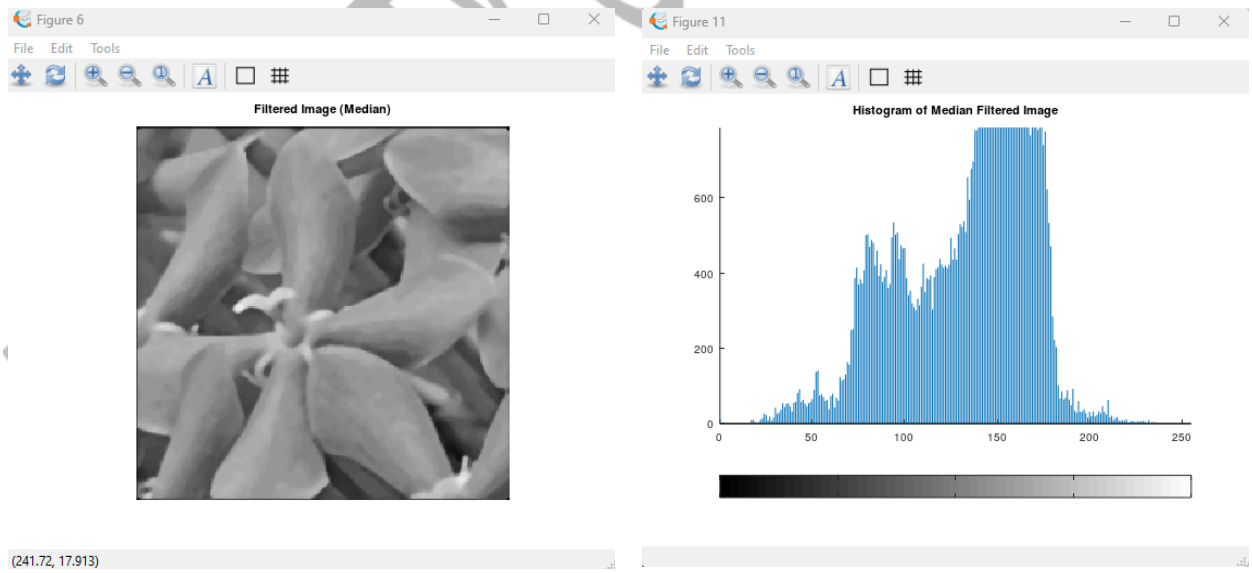


Python

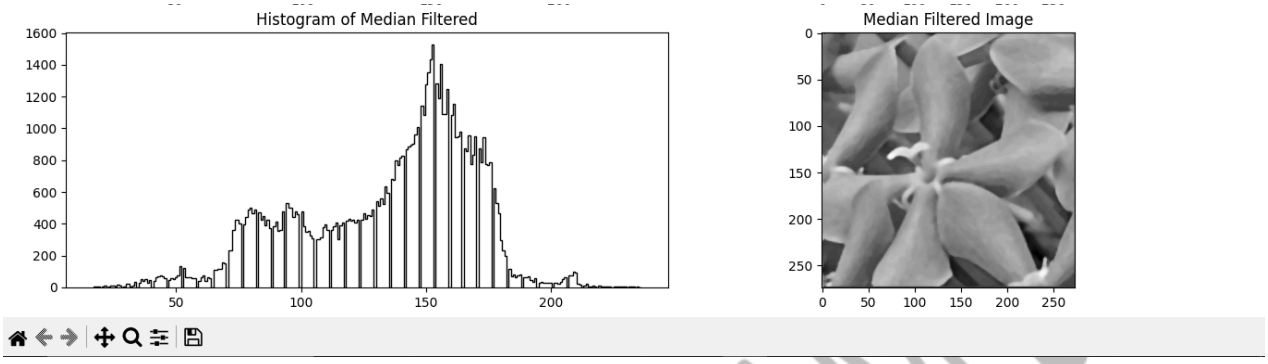
Figure 3: Histogram Equalized and Average Filtered Image and Its Histogram



Matlab



Octave



Python

Figure 4: Median Filtered Image and Its Histogram

These codes perform the following:

1. Grayscale conversion, which converts a color image (RGB) to a single-channel grayscale image. Colors are lost, but information about brightness is preserved. This depends on the desired outcome. If color information isn't crucial and you want to focus on brightness variations or prepare the image for further processing, grayscale conversion is effective. So in our image our original image is bright hence using the grayscale conversion is effective for our image that will be applied to other functions.
2. The Contrast Enhancement, which uses the function imadjust, stretches the contrast of the image by adjusting pixel values. Darker pixels become darker, and brighter pixels become brighter. This can make details in low-contrast areas more visible. The imadjust is effective for improving the visibility of features in images with low contrast. However, it can sometimes create an unnatural appearance or exaggerate noise in the image.
3. The Histogram Equalization uses the function histeq, which redistributes the pixel intensities in the image to create a flat histogram. This aims to achieve a more even distribution of brightness across the image. It is effective for images with uneven lighting or where specific features are obscured due to a concentration of pixels in a certain brightness range. It can enhance overall contrast and detail. However, it may sometimes create an overly artificial look or introduce artifacts.

4. Average filtering uses the function imfilter which replaces each pixel with the average value of its surrounding pixels which reduces noise in the image by blurring sharp edges and details. The average filter is effective for reducing random noise but can also blur important image features. It's good for removing minor noise while preserving larger structures.
5. Median filtering uses the function medfilt2 which replaces each pixel with the median value of its surrounding pixels. Similar to the average filter, it reduces noise but is less prone to blurring edges. It's particularly effective for removing salt-and-pepper noise (random black and white pixels). The median filter offers a good balance between noise reduction and edge preservation.

And lastly, each image uses a histogram through a function hist, which helps visualize the distribution of

pixel intensities. Visualizing histograms allows you to understand the original contrast distribution (grayscale) and how it's affected by the applied algorithms (contrast enhancement, equalization, filtering). This helps assess the effectiveness of each step.

Parameter Modification

<You can modify it to explore other functionalities>

```
% Convert to grayscale if the image is RGB
if size(img, 3) == 3
    img_gray = rgb2gray(img);
else
    img_gray = img;
end
% Filtering using average filter but different values
h_avg = fspecial('average', [10, 10]); % Original is [5,5]
img_avg_filtered = imfilter(img_gray, h_avg);

% Show the experimented image
figure;
imshow(img_avg_filtered);
title('Filtered Image (Using Average but Different values)');

% Filtering using median filter
img_median_filtered = medfilt2(img_gray, [1, 10]); % Original is [5,5]

% Display the median filtered image
figure;
imshow(img_median_filtered);
title('Experimented Filtered Image (Median)');

% Show the Histogram
figure;
imhist(img_median_filtered);
title('Histogram of Experimented Median Filtered)');
```

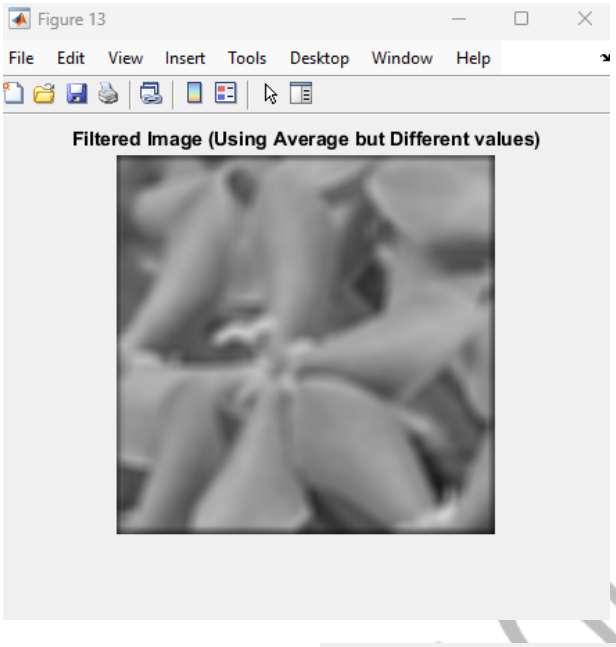


Figure 13
File Edit View Insert Tools Desktop Window Help
Filtered Image (Using Average but Different values)

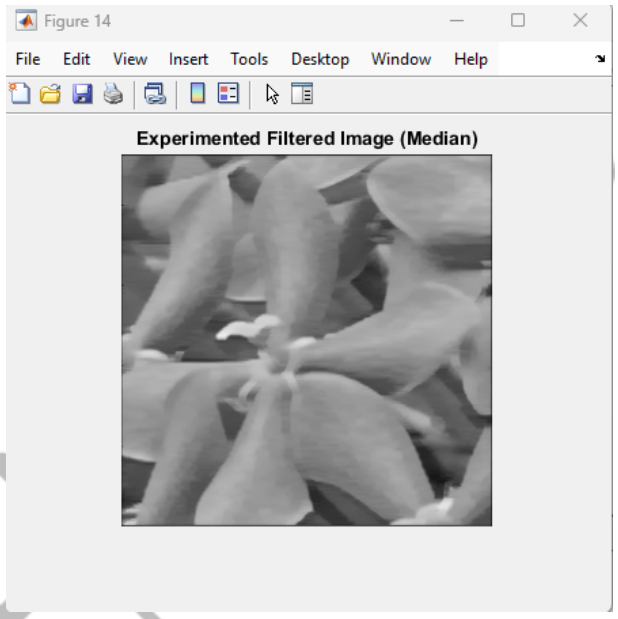


Figure 14
File Edit View Insert Tools Desktop Window Help
Experimented Filtered Image (Median)

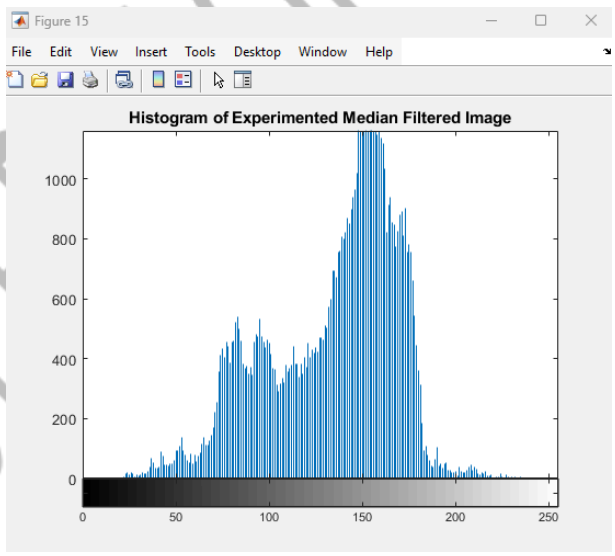
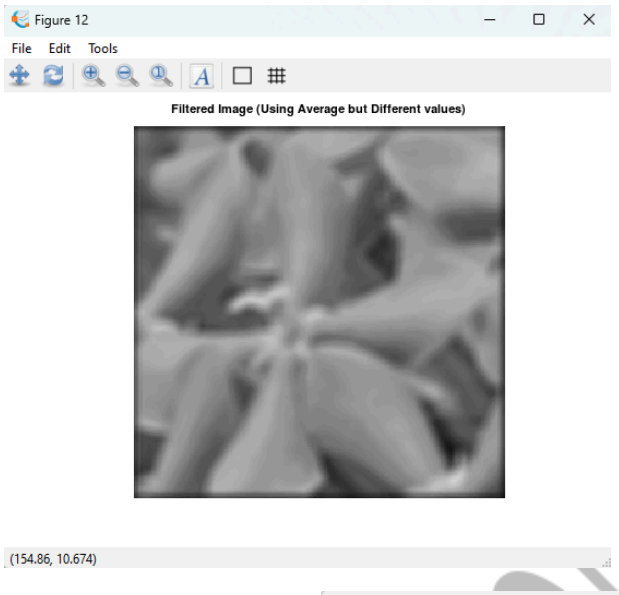
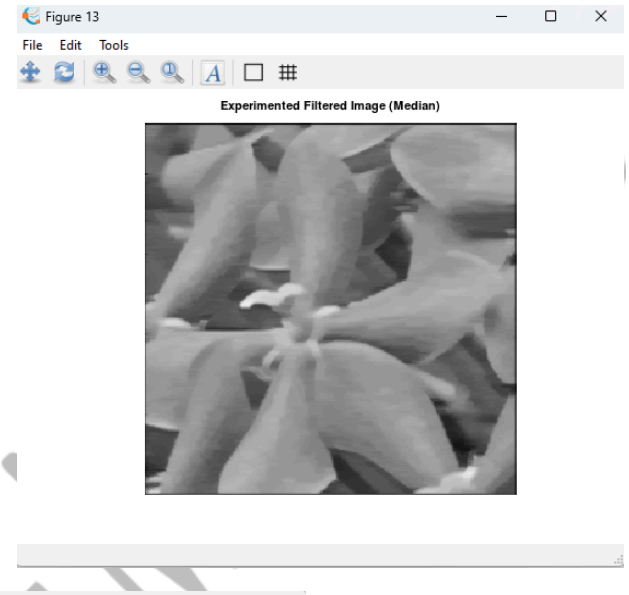
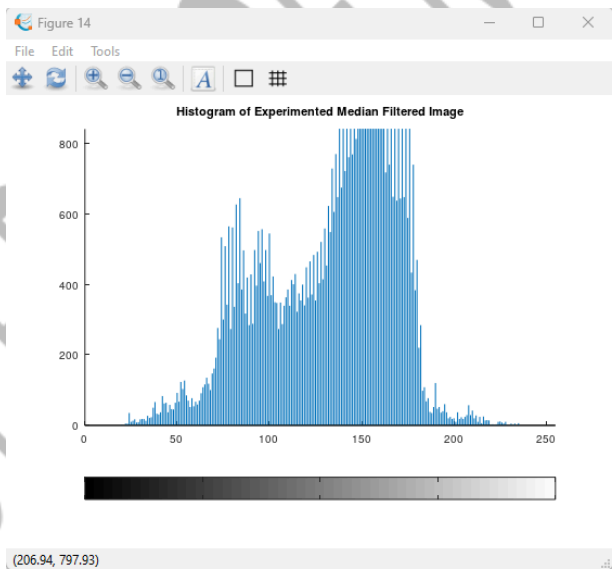


Figure 15
File Edit View Insert Tools Desktop Window Help
Histogram of Experimented Median Filtered Image
1000
800
600
400
200
0
0
50
100
150
200
250

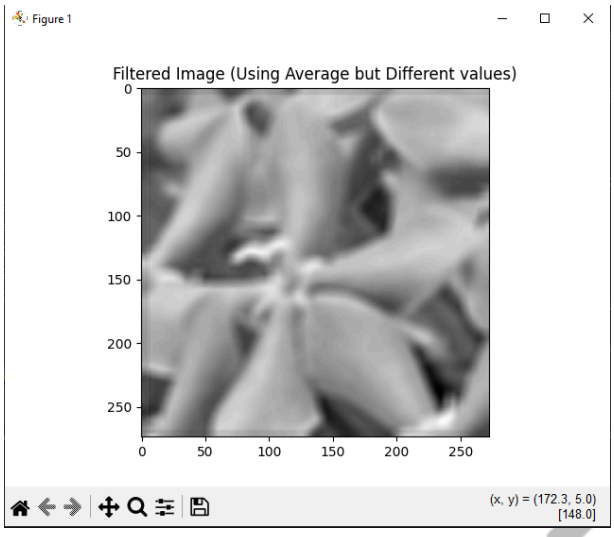
Matlab

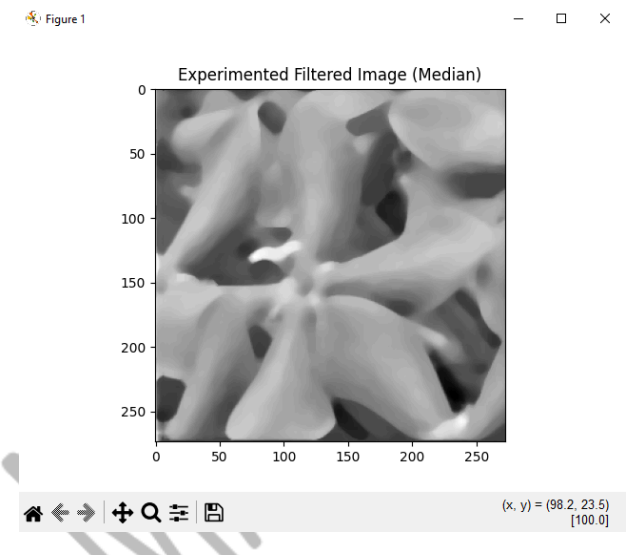


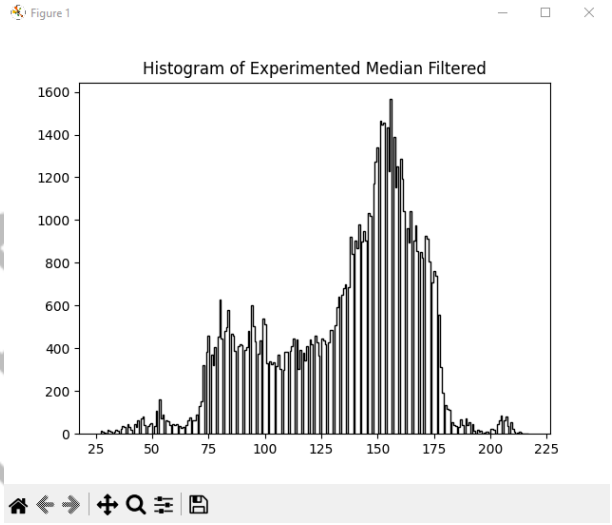




Octave







Python

Figure 5: Parameters Modification and Its Histogram

2. Visualize the results, analyze and interpret:

The applied algorithms resulted in noticeable changes to the image, which were clearly illustrated through visual comparisons of the original and processed images. Histograms provided a graphical representation of pixel intensity distributions, showcasing how each enhancement technique altered the image.

Grayscale conversion preserved brightness variations while simplifying the image, aiding in subsequent processing steps. Contrast enhancement using imadjust improved feature visibility but sometimes introduced noise, demonstrating the need for careful application.

Histogram equalization effectively redistributed pixel intensities, enhancing overall contrast and detail, though it could occasionally create an artificial appearance. Filtering techniques like average and median filtering reduced noise, with the former blurring edges and the latter preserving them, ensuring a balance between noise reduction and detail preservation.

IV. Conclusion

The laboratory activity successfully demonstrated the application of various image enhancement techniques using MATLAB and Python, highlighting their distinct effects on image quality. Through grayscale conversion, contrast enhancement, histogram equalization, and filtering, the exercise showcased how these methods can improve visibility, reduce noise, and enhance overall detail. The use of histograms as a visual tool effectively supported the analysis, allowing for a comprehensive evaluation of each algorithm's impact on the image.

References

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

<This is in a separate page>