LAB:03

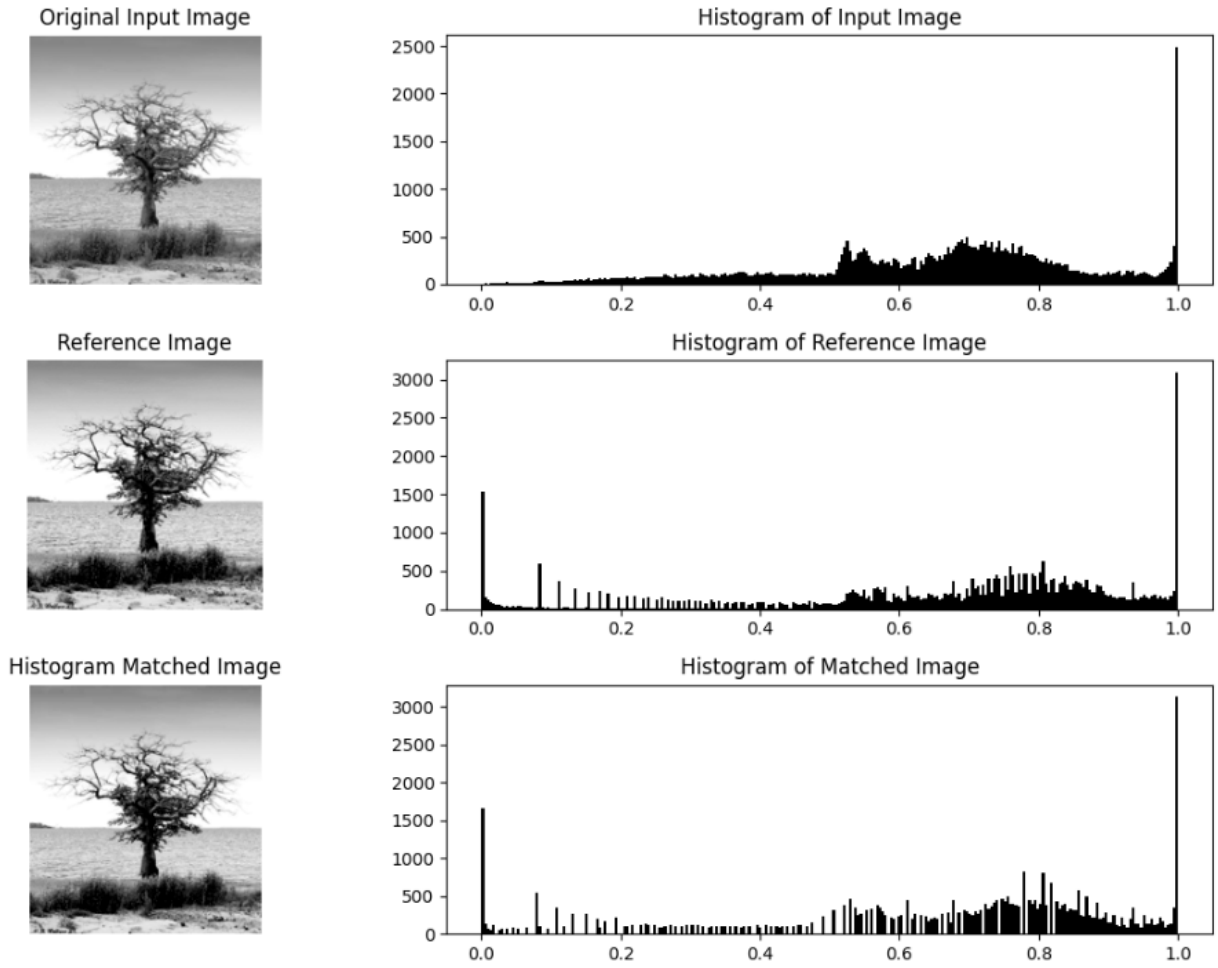
NAME: Urmi Kirtonia

ID:2022-1-60-184

```
import cv2
import numpy as np
import matplotlib.pyplot as plt
from skimage.measure import perimeter
from scipy.spatial import distance
def show_image(img, title ='', cmap = 'gray'):

   plt.imshow(img, cmap=cmap)
   plt.title(title)
   plt.axis('off')
   plt.show
fig1_path ='/kaggle/input/lab-184-003/lab3_cse438/Screenshot 2025-07-14 102639.png'
fig2_path ='/kaggle/input/lab-184-003/lab3_cse438/Screenshot 2025-07-14 103229.png'
fig3_path ='/kaggle/input/lab-184-003/lab3_cse438/Screenshot 2025-07-14 103104.png'
fig4_path ='/kaggle/input/lab-184-003/lab3_cse438/Screenshot 2025-07-14 103125.png'
fig5_path ='/kaggle/input/lab-184-003/lab3_cse438/Screenshot 2025-07-07 102450.png'
fig1 = cv2.imread(fig1_path,cv2.IMREAD_GRAYSCALE)
fig2 = cv2.imread(fig2_path,cv2.IMREAD_GRAYSCALE)
fig3 = cv2.imread(fig3_path,cv2.IMREAD_GRAYSCALE)
fig4 = cv2.imread(fig4_path,cv2.IMREAD_GRAYSCALE)
fig5 = cv2.imread(fig5_path,cv2.IMREAD_GRAYSCALE)
show_image(fig1,"Fig. 1")
show_image(fig2,"Fig. 2")
```

```
show_image(fig3,"Fig. 3")
show_image(fig4,"Fig. 4")
show_image(fig5,"Fig. 5")

1. from skimage import exposure
# Convert to float in [0, 1] for skimage
org_gray = fig1 / 255.0
ref_gray = fig2 / 255.0

# Histogram matching
matched_img = exposure.match_histograms(org_gray, ref_gray)

# Plotting
plt.figure(figsize=(12, 8))

plt.subplot(3, 2, 1)
plt.imshow(org_gray, cmap='gray')
plt.title('Original Input Image')
plt.axis('off')

plt.subplot(3, 2, 2)
plt.hist(org_gray.ravel(), bins=256, color='black')
plt.title('Histogram of Input Image')

plt.subplot(3, 2, 3)
plt.imshow(ref_gray, cmap='gray')
```

```
plt.title('Reference Image')
plt.axis('off')

plt.subplot(3, 2, 4)
plt.hist(ref_gray.ravel(), bins=256, color='black')
plt.title('Histogram of Reference Image')

plt.subplot(3, 2, 5)
plt.imshow(matched_img, cmap='gray')
plt.title('Histogram Matched Image')
plt.axis('off')

plt.subplot(3, 2, 6)
plt.hist(matched_img.ravel(), bins=256, color='black')
plt.title('Histogram of Matched Image')

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

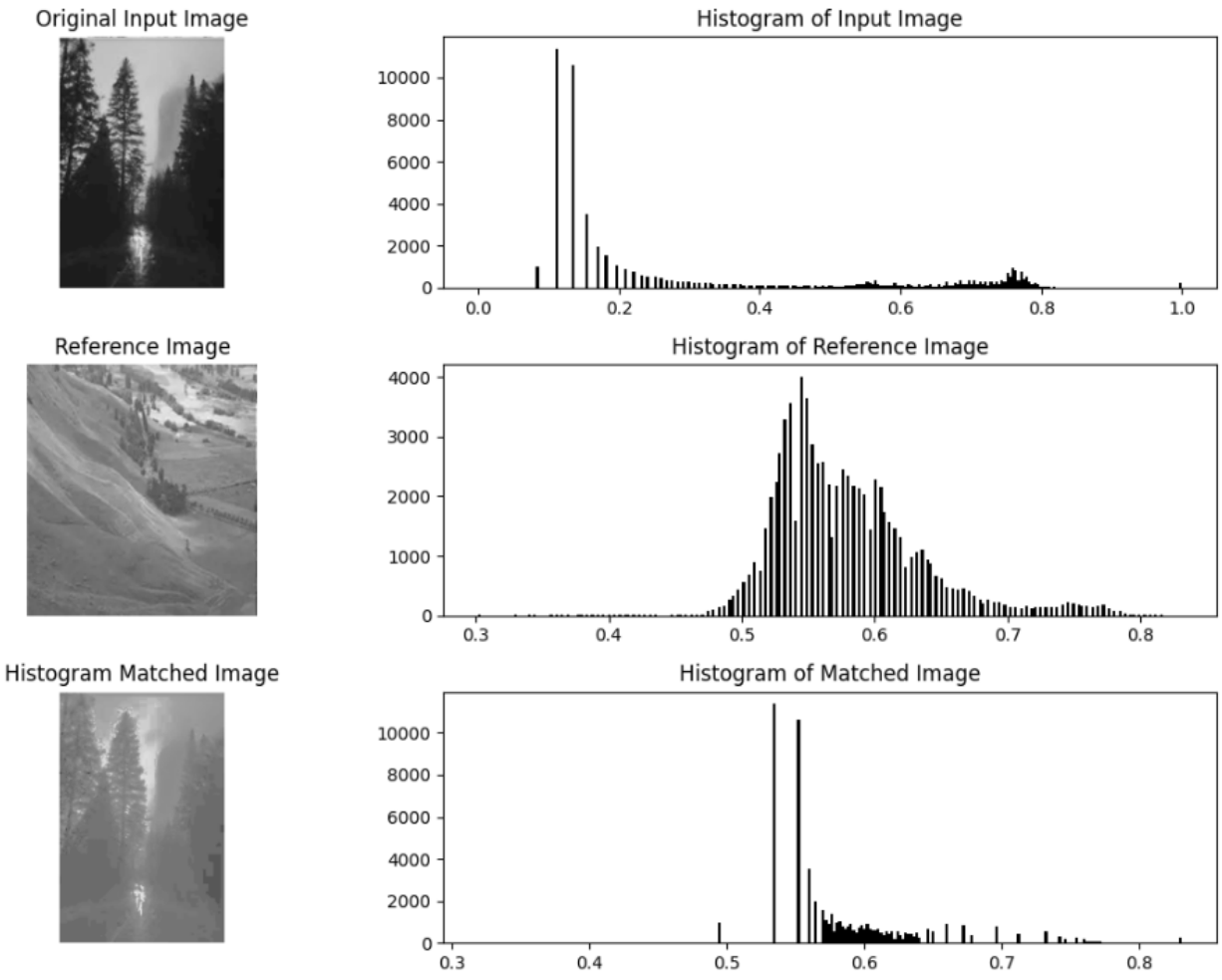



```
# Convert to float in [0, 1] for skimage
org_gray = fig3 / 255.0
ref_gray = fig4 / 255.0

# Histogram matching
matched_img = exposure.match_histograms(org_gray, ref_gray)

# Plotting
plt.figure(figsize=(12, 8))
```

```
plt.subplot(3, 2, 1)
plt.imshow(org_gray, cmap='gray')
plt.title('Original Input Image')
plt.axis('off')

plt.subplot(3, 2, 2)
plt.hist(org_gray.ravel(), bins=256, color='black')
plt.title('Histogram of Input Image')

plt.subplot(3, 2, 3)
plt.imshow(ref_gray, cmap='gray')
plt.title('Reference Image')
plt.axis('off')

plt.subplot(3, 2, 4)
plt.hist(ref_gray.ravel(), bins=256, color='black')
plt.title('Histogram of Reference Image')

plt.subplot(3, 2, 5)
plt.imshow(matched_img, cmap='gray')
plt.title('Histogram Matched Image')
plt.axis('off')

plt.subplot(3, 2, 6)
plt.hist(matched_img.ravel(), bins=256, color='black')
plt.title('Histogram of Matched Image')
```

plt.tight_layout()

plt.show()



2. import matplotlib.pyplot as plt

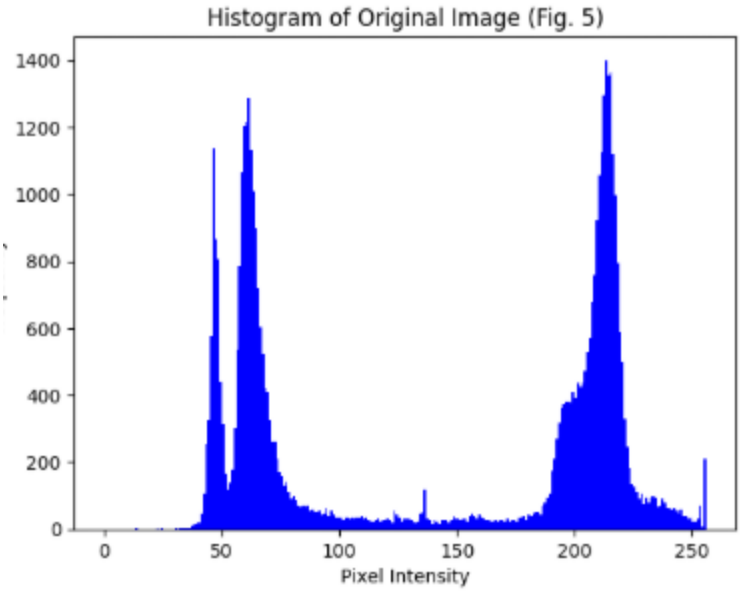
Apply histogram equalization

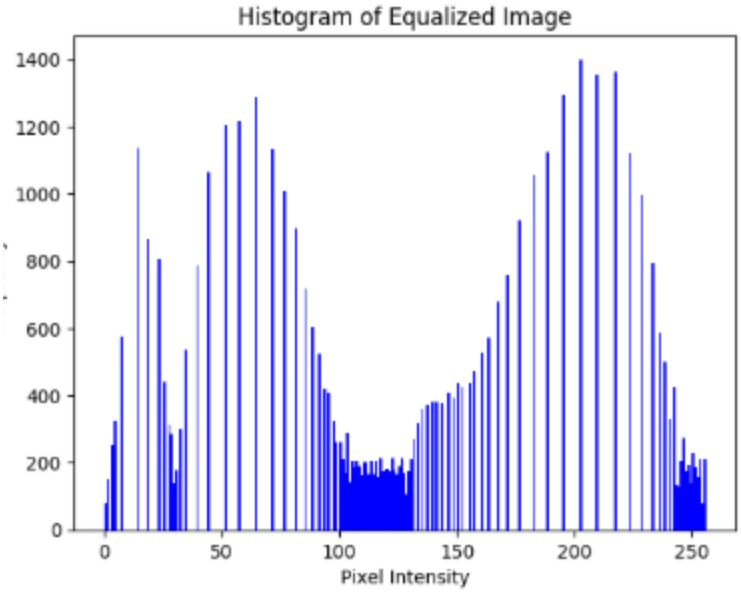
equalized_fig5 = cv2.equalizeHist(fig5)

```
# Function to show image
def show_image(img, title):
    plt.imshow(img, cmap='gray')
    plt.title(title)
    plt.axis('off')
    plt.show()

# Function to plot histogram
def plot_histogram(img, title):
    plt.hist(img.ravel(), bins=256, range=(0, 256), color='blue')
    plt.title(title)
    plt.xlabel('Pixel Intensity')
    plt.ylabel('Frequency')
    plt.show()

# Show original and equalized images
show_image(fig5, "Original Image (Fig. 5)")
show_image(equalized_fig5, "Histogram Equalized Image")

# Plot histograms
plot_histogram(fig5, "Histogram of Original Image (Fig. 5)")
plot_histogram(equalized_fig5, "Histogram of Equalized Image")
```

Original Image (Fig. 5)

Histogram Equalized Image


Histogram of Original Image (Fig. 5)
1400
1200
1000
800
600
400
200
0
0
50
100
150
200
250
Pixel Intensity


Histogram of Equalized Image
1400
1200
1000
800
600
400
200
0
0
50
100
150
200
250
Pixel Intensity