EAST WEST UNIVERSITY

CSE438

Section: 01

Lab: 03 Report

Topic: Match the reference image using histogram matching

Submitted By:

Name: Md Sifat Ullah Sheikh

ID: 2022-1-60-029

Submitted To:

K.M. Safin Kamal

Lecturer

Department of Computer Science & Engineering

Date: 14 July 2025

Q1. Adjust the histogram of the following image to match the reference image using histogram matching. Show the histogram of original, reference, and output images.

CODE:

```
 import cv2
import matplotlib.pyplot as plt
from skimage.exposure import match_histograms


src_path = '/kaggle/input/cse-438-lab-03/CSE_438_LAB_03/Picture3.png'
ref_path = '/kaggle/input/cse-438-lab-03/CSE_438_LAB_03/Picture4.png'


src_rgb = cv2.cvtColor(cv2.imread(src_path), cv2.COLOR_BGR2RGB)
ref_rgb = cv2.cvtColor(cv2.imread(ref_path), cv2.COLOR_BGR2RGB)


matched_rgb = match_histograms(src_rgb, ref_rgb, channel_axis=-1)


def plot_rgb_histogram(image, ax, title):
    color = ('r', 'g', 'b')
    for i, col in enumerate(color):
        hist = cv2.calcHist([image], [i], None, [256], [0,256])
        ax.plot(hist, color=col)
    ax.set_title(title)
    ax.set_xlim([0, 256])
    ax.set_xlabel('Pixel Value')
    ax.set_ylabel('Frequency')


plt.figure(figsize=(18, 5))
titles = ['Source Image', 'Reference Image', 'Matched Image']
images = [src_rgb, ref_rgb, matched_rgb]

for i in range(3):
    plt.subplot(1, 3, i+1)
    plt.imshow(images[i])
    plt.title(titles[i])
    plt.axis('off')
plt.tight_layout()
plt.show()
```

```
fig, axes = plt.subplots(1, 3, figsize=(18, 4))
hist_titles = ['Source Histogram', 'Reference Histogram', 'Matched Histogram']
for i, ax in enumerate(axes):
    plot_rgb_histogram(images[i], ax, hist_titles[i])

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

```
import cv2
import matplotlib.pyplot as plt
from skimage.exposure import match_histograms


src_path = '/kaggle/input/cse-438-lab-03/CSE_438_LAB_03/Picture3.png'
ref_path = '/kaggle/input/cse-438-lab-03/CSE_438_LAB_03/Picture4.png'


src_rgb = cv2.cvtColor(cv2.imread(src_path), cv2.COLOR_BGR2RGB)
ref_rgb = cv2.cvtColor(cv2.imread(ref_path), cv2.COLOR_BGR2RGB)
```

```
matched_rgb = match_histograms(src_rgb, ref_rgb, channel_axis=-1)


def plot_rgb_histogram(image, ax, title):
    color = ('r', 'g', 'b')
    for i, col in enumerate(color):
        hist = cv2.calcHist([image], [i], None, [256], [0,256])
        ax.plot(hist, color=col)
    ax.set_title(title)
    ax.set_xlim([0, 256])
    ax.set_xlabel('Pixel Value')
    ax.set_ylabel('Frequency')


plt.figure(figsize=(18, 5))
titles = ['Source Image', 'Reference Image', 'Matched Image']
images = [src_rgb, ref_rgb, matched_rgb]

for i in range(3):
    plt.subplot(1, 3, i+1)
    plt.imshow(images[i])
    plt.title(titles[i])
    plt.axis('off')
plt.tight_layout()
plt.show()


fig, axes = plt.subplots(1, 3, figsize=(18, 4))
hist_titles = ['Source Histogram', 'Reference Histogram', 'Matched Histogram']
for i, ax in enumerate(axes):
    plot_rgb_histogram(images[i], ax, hist_titles[i])

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

Q2. Change the contrast of the image using histogram equalization. Show the histogram of both input and output images.

CODE

```
 import cv2
import matplotlib.pyplot as plt


img_path = '/kaggle/input/cse-438-lab-03/CSE_438_LAB_03/Picture5.png'


img_bgr = cv2.imread(img_path)


img_ycrcb = cv2.cvtColor(img_bgr, cv2.COLOR_BGR2YCrCb)
```

```
y, cr, cb = cv2.split(img_ycrcb)


y_eq = cv2.equalizeHist(y)


img_ycrcb_eq = cv2.merge([y_eq, cr, cb])
img_bgr_eq = cv2.cvtColor(img_ycrcb_eq, cv2.COLOR_YCrCb2BGR)


img_rgb = cv2.cvtColor(img_bgr, cv2.COLOR_BGR2RGB)
img_rgb_eq = cv2.cvtColor(img_bgr_eq, cv2.COLOR_BGR2RGB)


plt.figure(figsize=(12, 5))

plt.subplot(1, 2, 1)
plt.imshow(img_rgb)
plt.title('Original Color Image')
plt.axis('off')

plt.subplot(1, 2, 2)
plt.imshow(img_rgb_eq)
plt.title('Equalized Color Image')
plt.axis('off')

plt.tight_layout()
plt.show()

# Save the equalized color image
cv2.imwrite('/kaggle/working/color_equalized_image.png', img_bgr_eq)
print("☑ Color equalized image saved at: /kaggle/working/color_equalized_image.png")
```

```
def plot_rgb_hist(image, ax, title):
    colors = ('r', 'g', 'b')
    for i, col in enumerate(colors):
        hist = cv2.calcHist([image], [i], None, [256], [0, 256])
        ax.plot(hist, color=col)
    ax.set_title(title)
    ax.set_xlim([0, 256])
    ax.set_xlabel("Pixel Value")
    ax.set_ylabel("Frequency")


fig, axs = plt.subplots(1, 2, figsize=(14, 4))
plot_rgb_hist(img_rgb, axs[0], "Original RGB Histogram")
plot_rgb_hist(img_rgb_eq, axs[1], "Equalized RGB Histogram")
plt.tight_layout()
plt.show()


cv2.imwrite('/kaggle/working/color_equalized_image.png', img_bgr_eq)
print("☑ Color equalized image saved at: /kaggle/working/color_equalized_image.png")
```

Original RGB Histogram
Frequency
Pixel Value
0
200
400
600
800
1000
1200
1400
1600
0
50
100
150
200
250

Equalized RGB Histogram
Frequency
Pixel Value
0
200
400
600
800
1000
1200
1400
1600
0
50
100
150
200
250