MODUL 3

PENGOLAHAN CITRA DIGITAL
MANIPULASI HISTOGRAM CITRA

D3/D4 TEKNIK INFORMATIKA
JURUSAN TEKNIK KOMPUTER DAN INFORMATIKA
POLITEKNIK NEGERI BANDUNG

MAHASISWA 043 | PENGOLAHAN CITRA DIGITAL | SEPTEMBER, 4 2023

CLUE

1. Anda dapat menduplikat file tugas sebelumnya.
2. import matplotlib.pyplot as plt (jika belum ada matplotlib silakan di pip install).
3. Pada praktikum ini anda hanya diberikan clue code,

```
b, g, r = cv2.split(img)
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
# Calculate the histogram of each channel
hist_b = cv2.calcHist([b], [0], None, [256], [0, 256])
hist_g = cv2.calcHist([g], [0], None, [256], [0, 256])
hist_r = cv2.calcHist([r], [0], None, [256], [0, 256])
hist_gray = cv2.calcHist([gray], [0], None, [256], [0, 256])
hist = cv2.calcHist([img], [0, 1, 2], None, [8, 8, 8], [0, 256, 0, 256, 0, 256])
# Plot the histograms
plt.plot(hist_b, color = 'blue')
plt.plot(hist_g, color = 'green')
plt.plot(hist_r, color = 'red')
plt.plot(hist_gray,color = ‘black’)
plt.xlim([0, 256])
plt.imshow(hist)
plt.show()

#average_intensity
average_intensity = np.mean(gray)
#image contrast
def calculateContrast(img):
    img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    std = np.std(img)
    return std
contrast = calculateContrast(img)
print("Contrast:", contrast)
```

TASK PRAKTIKUM

TASK 0: HAI HISTOGRAM CITRA

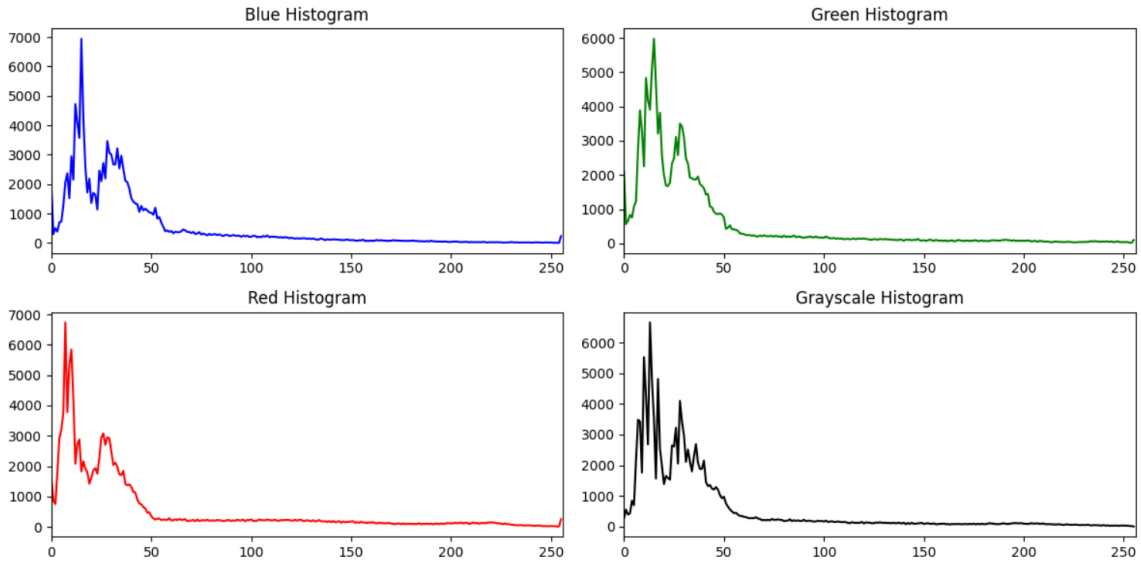
1. Buat dan jelaskan
    a. histogram rgb,
    Histogram RGB adalah representasi statistik dari distribusi intensitas warna merah (R), hijau (G), dan biru (B) dalam sebuah gambar. Histogram ini membantu dalam menganalisis karakteristik warna gambar, seperti kontras warna, distribusi warna, dan tingkat kecerahan.

    Histogram RGB biasanya memiliki tiga saluran terpisah, satu untuk setiap komponen warna (R, G, dan B). Setiap saluran memiliki sumbu x yang mewakili intensitas piksel dari 0 (tidak ada warna) hingga 255 (intensitas maksimum). Suma intensitas piksel dalam setiap saluran warna dihitung dan diplot sebagai histogram.

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

# Load the image
img = cv2.imread('foto.jpeg')

# Split the image into BGR channels
b, g, r = cv2.split(img)

# Convert the image to grayscale
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# Calculate the histogram of each channel
hist_b = cv2.calcHist([b], [0], None, [256], [0, 256])
hist_g = cv2.calcHist([g], [0], None, [256], [0, 256])
hist_r = cv2.calcHist([r], [0], None, [256], [0, 256])
hist_gray = cv2.calcHist([gray], [0], None, [256], [0,
256])

# Plot the histograms
plt.figure(figsize=(12, 6))

plt.subplot(2, 2, 1)
plt.plot(hist_b, color='blue')
plt.title('Blue Histogram')
plt.xlim([0, 256])

plt.subplot(2, 2, 2)
plt.plot(hist_g, color='green')
plt.title('Green Histogram')
plt.xlim([0, 256])

plt.subplot(2, 2, 3)
plt.plot(hist_r, color='red')
plt.title('Red Histogram')
plt.xlim([0, 256])

plt.subplot(2, 2, 4)
plt.plot(hist_gray, color='black')
plt.title('Grayscale Histogram')
plt.xlim([0, 256])

plt.tight_layout()

# Display the histograms
plt.show()

# Calculate average intensity
average_intensity = np.mean(gray)
print("Average Intensity:", average_intensity)
```

```
    # Calculate image contrast
    def calculateContrast(img):
        img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        std = np.std(img)
        return std

    contrast = calculateContrast(img)
print("Contrast:", contrast)
```

Output



b. histogram grayscale,

Histogram grayscale adalah representasi visual dari distribusi intensitas piksel dalam gambar hitam putih (grayscale). Histogram ini menggambarkan seberapa sering intensitas piksel tertentu muncul di seluruh gambar. Setiap sumbu pada histogram grayscale menggambarkan rentang intensitas piksel, sementara sumbu vertikal menunjukkan jumlah piksel dengan intensitas tertentu di rentang tersebut.

```
    import cv2
    import numpy as np
    import matplotlib.pyplot as plt

    # Load the grayscale image
    gray_img = cv2.imread('foto.jpeg', cv2.IMREAD_GRAYSCALE)

    # Calculate the histogram of the grayscale image
    hist_gray = cv2.calcHist([gray_img], [0], None, [256], [0,
    256])

    # Plot the histogram
    plt.plot(hist_gray, color='black')
    plt.title('Grayscale Histogram')
    plt.xlim([0, 256])
    plt.xlabel('Pixel Value')
    plt.ylabel('Frequency')

    # Display the histogram
```

```
plt.show()
```

c. average intensity,

Average intensity (intensitas rata-rata) adalah nilai rata-rata dari intensitas piksel dalam sebuah gambar atau wilayah tertentu dalam gambar. Intensitas piksel mengukur tingkat kecerahan atau warna pada lokasi piksel tersebut dalam gambar.

```
import cv2
import numpy as np

# Load the image
img = cv2.imread('foto.jpeg')

# Convert the image to grayscale
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# Calculate average intensity
average_intensity = np.mean(gray)

# Display the average intensity
print("Average Intensity:", average_intensity)
```

✓ 0.0s

```
Average Intensity: 41.33989197530864
```

d. image contrast

Kontras dalam konteks gambar mengacu pada perbedaan antara intensitas berbagai elemen visual dalam gambar, seperti perbedaan antara kecerahan dan warna antara objek atau detail yang berbeda. Semakin besar perbedaan antara intensitas elemen-elemen ini, semakin tinggi kontrasnya. Kontras adalah salah satu elemen yang penting dalam desain visual dan fotografi karena dapat mempengaruhi bagaimana

gambar tersebut terlihat dan mudah dipahami oleh mata manusia.

```python
import cv2
import numpy as np

# Load the image
img = cv2.imread('foto.jpeg')

# Convert the image to grayscale
gray = cv2.cvtColor(img, cv2.COLO

# Calculate image contrast
def calculateContrast(img):
    std = np.std(img)
    return std

contrast = calculateContrast(gray

# Display the contrast
print("Contrast:", contrast)
```

```
✓ 0.0s
Contrast: 45.72130497344583
```

TASK 1: AHLI HISTOGRAM CITRA

1. Buat fungsi Histogram Spesicification dan Histogram equilization, dan jelaskan perbedaan kedua fungsi tersebut.
   Histogram Spesification :
   Histogram Specification dan Histogram Equalization adalah dua teknik yang digunakan dalam pengolahan gambar untuk memanipulasi histogram sebuah gambar. Mereka memiliki perbedaan dalam cara mereka mengubah distribusi intensitas piksel dalam gambar dan tujuan akhir mereka. Berikut adalah perbedaan antara keduanya:

   Tujuan Utama:

   Histogram Equalization: Tujuan utama dari Histogram Equalization adalah untuk mengubah histogram gambar agar mendekati distribusi intensitas piksel yang merata, sehingga meningkatkan kontras gambar secara keseluruhan. Ini sering digunakan untuk meningkatkan kualitas visual gambar yang memiliki kontras yang rendah atau distribusi intensitas yang tidak merata.
   Histogram Specification: Histogram Specification, juga dikenal sebagai Histogram Matching, memiliki tujuan khusus untuk mencocokkan histogram gambar target tertentu. Ini digunakan ketika Anda ingin mengubah histogram gambar sumber menjadi histogram yang sesuai dengan gambar target, sehingga menghasilkan gambar yang memiliki karakteristik histogram tertentu.
   Proses:

   Histogram Equalization: Histogram Equalization melibatkan penghitungan kumulatif distribusi intensitas piksel dalam gambar sumber dan kemudian mengganti setiap nilai piksel dengan nilai yang sesuai dalam histogram yang sudah disesuaikan. Hasilnya adalah gambar dengan distribusi intensitas yang lebih merata.

Histogram Specification: Histogram Specification melibatkan perbandingan histogram gambar sumber dengan histogram gambar target yang diinginkan. Kemudian, berdasarkan perbandingan ini, setiap nilai piksel dalam gambar sumber diganti sehingga cocok dengan histogram target.
Penggunaan:

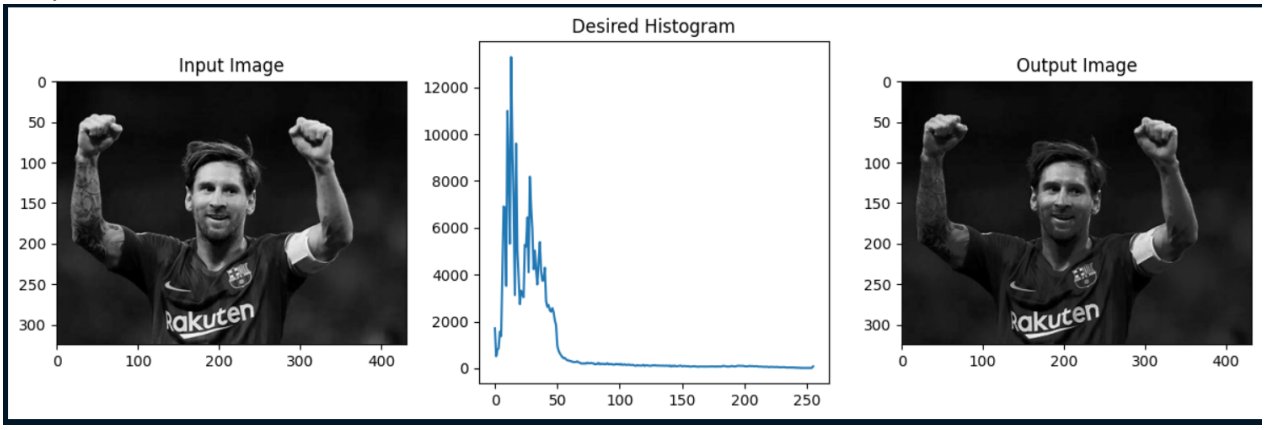
Histogram Equalization: Histogram Equalization digunakan untuk meningkatkan kontras gambar secara umum. Ini dapat digunakan di berbagai aplikasi pengolahan gambar.
Histogram Specification: Histogram Specification digunakan ketika Anda memiliki gambar target yang ingin Anda capai dalam hal distribusi intensitas, seperti dalam kasus medis imaging atau ketika Anda ingin mencocokkan gambar dengan referensi tertentu.
Kontrol:

Histogram Equalization: Histogram Equalization memiliki sedikit kontrol terhadap hasil akhir, dan perubahan yang dihasilkan mungkin tidak sesuai dengan preferensi subjektif.
Histogram Specification: Histogram Specification memberikan lebih banyak kontrol karena Anda dapat menentukan gambar target histogram yang diinginkan, sehingga menghasilkan gambar yang lebih sesuai dengan preferensi Anda.
Kedua teknik ini memiliki peran yang berbeda dalam pengolahan gambar, dan pemilihan teknik yang tepat tergantung pada tujuan dan kebutuhan spesifik Anda dalam mengubah distribusi intensitas piksel dalam gambar.

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

def histogram_specification(input_image, desired_histogram):
    # Mengubah citra input menjadi citra grayscale jika belum
    if len(input_image.shape) == 3:
        input_image = cv2.cvtColor(input_image,
cv2.COLOR_BGR2GRAY)

    # Menghitung histogram dari citra input
    hist_image = cv2.calcHist([input_image], [0], None, [256],
[0, 256])

    # Menghitung kumulatif histogram dari citra input
    cdf_image = hist_image.cumsum()
    cdf_image_normalized = cdf_image / cdf_image[-1]

    # Menghitung kumulatif histogram dari histogram yang
diinginkan
    cdf_desired = desired_histogram.cumsum()
    cdf_desired_normalized = cdf_desired / cdf_desired[-1]

    # Mencocokkan citra dengan histogram yang diinginkan
    mapping = np.interp(cdf_image_normalized,
cdf_desired_normalized, range(256))
    output_image = mapping[input_image]

    return output_image.astype(np.uint8)

# Contoh penggunaan:
```

```
if __name__ == "__main__":
    # Membaca citra input
    input_image = cv2.imread("foto.jpeg", cv2.IMREAD_GRAYSCALE)

    # Menghitung histogram dari citra input
    hist_desired = cv2.calcHist([input_image], [0], None,
[256], [0, 256])

    # Mengubah citra histogram yang diinginkan (contoh:
menaikkan kontras)
    hist_desired_modified = np.copy(hist_desired)
    hist_desired_modified[0:50] *= 2  # Contoh: meningkatkan
kontras di rentang intensitas 100-200

    # Mengaplikasikan Histogram Specification
    output_image = histogram_specification(input_image,
hist_desired_modified)

    # Menampilkan citra input, citra histogram yang diinginkan,
dan citra hasil
    plt.figure(figsize=(12, 4))

    plt.subplot(131)
    plt.imshow(input_image, cmap='gray')
    plt.title("Input Image")

    plt.subplot(132)
    plt.plot(hist_desired_modified)
    plt.title("Desired Histogram")

    plt.subplot(133)
    plt.imshow(output_image, cmap='gray')
    plt.title("Output Image")

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

Output :

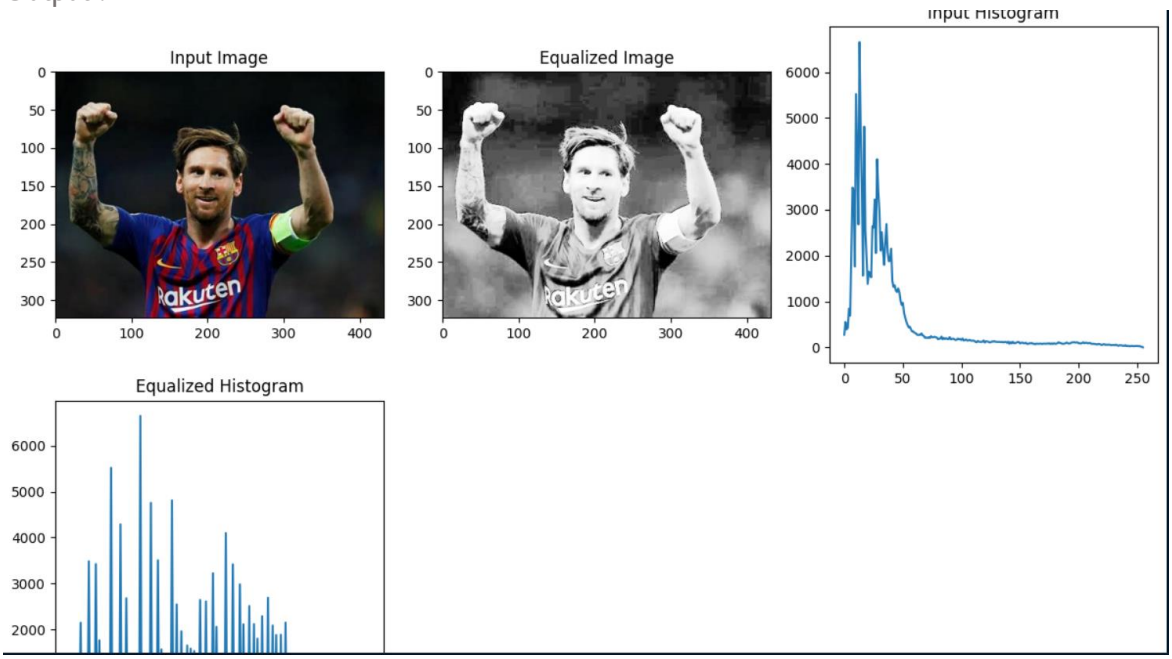


Histogram equalization :

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

def histogram_equalization(input_image):
    # Mengubah citra input menjadi citra grayscale jika belum
    if len(input_image.shape) == 3:
        input_image = cv2.cvtColor(input_image,
cv2.COLOR_BGR2GRAY)

    # Melakukan ekualisasi histogram
    equalized_image = cv2.equalizeHist(input_image)

    # Menghitung histogram dari citra input dan citra hasil
    hist_input = cv2.calcHist([input_image], [0], None, [256],
[0, 256])
    hist_equalized = cv2.calcHist([equalized_image], [0], None,
[256], [0, 256])

    return equalized_image, hist_input, hist_equalized

# Contoh penggunaan:
if __name__ == "__main__":
    # Membaca citra input
    input_image = cv2.imread("foto.jpeg")

    # Mengaplikasikan Histogram Equalization dan menghitung
histogramnya
    equalized_image, hist_input, hist_equalized =
histogram_equalization(input_image)

    # Menampilkan citra input, histogram, dan citra hasil di
tengah
    plt.figure(figsize=(12, 8))

    plt.subplot(231)
    plt.imshow(cv2.cvtColor(input_image, cv2.COLOR_BGR2RGB))
    plt.title("Input Image")

    plt.subplot(232)
    plt.imshow(equalized_image, cmap='gray')
    plt.title("Equalized Image")

    plt.subplot(233)
    plt.plot(hist_input)
    plt.title("Input Histogram")

    plt.subplot(234)
    plt.plot(hist_equalized)
```

```
        plt.title("Equalized Histogram")

        plt.subplot(235)
        plt.axis('off')  # Menonaktifkan sumbu pada gambar
    histogram kosong
        plt.title("")

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

Output :



Perbandingan :

```python
import cv2
import numpy as np
import matplotlib.pyplot as plt

def histogram_specification(input_image, desired_histogram):
    # ... (Kode histogram specification seperti yang telah dibahas sebelumnya) ...
    if len(input_image.shape) == 3:
        input_image = cv2.cvtColor(input_image, cv2.COLOR_BGR2GRAY)

    # Menghitung histogram dari citra input
    hist_image = cv2.calcHist([input_image], [0], None, [256], [0, 256])

    # Menghitung kumulatif histogram dari citra input
    cdf_image = hist_image.cumsum()
    cdf_image_normalized = cdf_image / cdf_image[-1]

    # Menghitung kumulatif histogram dari histogram yang diinginkan
    cdf_desired = desired_histogram.cumsum()
```

```
    cdf_desired_normalized = cdf_desired / cdf_desired[-1]

    # Mencocokkan citra dengan histogram yang diinginkan
    mapping = np.interp(cdf_image_normalized, cdf_desired_normalized, range(256))
    output_image = mapping[input_image]

    return output_image.astype(np.uint8)

def histogram_equalization(input_image):
    # Mengubah citra input menjadi citra grayscale jika belum
    if len(input_image.shape) == 3:
        input_image = cv2.cvtColor(input_image, cv2.COLOR_BGR2GRAY)

    # Melakukan ekualisasi histogram
    equalized_image = cv2.equalizeHist(input_image)

    # Menghitung histogram dari citra input dan citra hasil
    hist_input = cv2.calcHist([input_image], [0], None, [256], [0, 256])
    hist_equalized = cv2.calcHist([equalized_image], [0], None, [256], [0, 256])

    return equalized_image, hist_input, hist_equalized

# Contoh penggunaan:
if __name__ == "__main__":
    # Membaca citra input
    input_image = cv2.imread("foto.jpeg")

    # Menghitung histogram dari citra input
    hist_desired = cv2.calcHist([input_image], [0], None, [256], [0, 256])

    # Mengubah citra histogram yang diinginkan (contoh: menaikkan kontras)
    hist_desired_modified = np.copy(hist_desired)
    hist_desired_modified[100:200] *= 2  # Contoh: meningkatkan kontras di rentang
intensitas 100-200

    # Mengaplikasikan Histogram Specification
    output_image_specification = histogram_specification(input_image,
hist_desired_modified)

    # Mengaplikasikan Histogram Equalization
    output_image_equalization, hist_input, hist_equalized =
histogram_equalization(input_image)

    # Menampilkan gambar asli, gambar hasil specification, gambar hasil equalization,
serta histogramnya
    plt.figure(figsize=(16, 12))

    plt.subplot(231)
    plt.imshow(cv2.cvtColor(input_image, cv2.COLOR_BGR2RGB))
```

```
    plt.title("Original Image")

    plt.subplot(232)
    plt.imshow(output_image_specification, cmap='gray')
    plt.title("Specification Image")

    plt.subplot(233)
    plt.imshow(output_image_equalization, cmap='gray')
    plt.title("Equalization Image")

    plt.subplot(234)
    plt.plot(hist_desired_modified)
    plt.title("Desired Histogram")

    plt.subplot(235)
    plt.plot(hist_input)
    plt.title("Input Histogram")

    plt.subplot(236)
    plt.plot(hist_equalized)
    plt.title("Equalized Histogram")

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

TASK 2: LESSON LEARNT

2. Tulis Lesson Learnt dari praktikum ini, Lesson learnt ditulis tangan.