TUGAS PENGOLAHAN CITRA DIGITAL

Disusun Untuk Memenuhi Salah Satu Tugas Mata Kuliah Pengolahan Citra Digital

Dosen Pengampu : Leni Fitriani, ST., M.Kom.

Disusun oleh :

Melina Amelia

2206152

Informatika -E

PROGRAM STUDI TEKNIK INFORMATIKA

INSTITUT TEKNOLOGI GARUT

2024

A. Analisis Kode Program Pertama

1. Kode Program

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

# Step 1: Load an image (Replace the path with the actual image file path)
image_path = '/content/drive/MyDrive/Pengolahan Citra Digital/Melina.jpg'
# Ganti dengan path gambar Anda
image = cv2.imread(image_path)

# Step 2: Convert the image from BGR (OpenCV default) to RGB
image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

# Step 3: Split the image into its Red, Green, and Blue channels
R, G, B = cv2.split(image_rgb)

# Step 4: Create a zero matrix to help in visualization
zeros = np.zeros(image.shape[:2], dtype="uint8")

# Step 5: Visualize the R, G, B channels separately
plt.figure(figsize=(10, 10))

# Red channel
plt.subplot(2, 2, 1)
red_image = cv2.merge([R, zeros, zeros])  # R channel with G, B as 0
plt.imshow(red_image)
plt.title('Red Channel')
plt.axis('off')

# Green channel
```

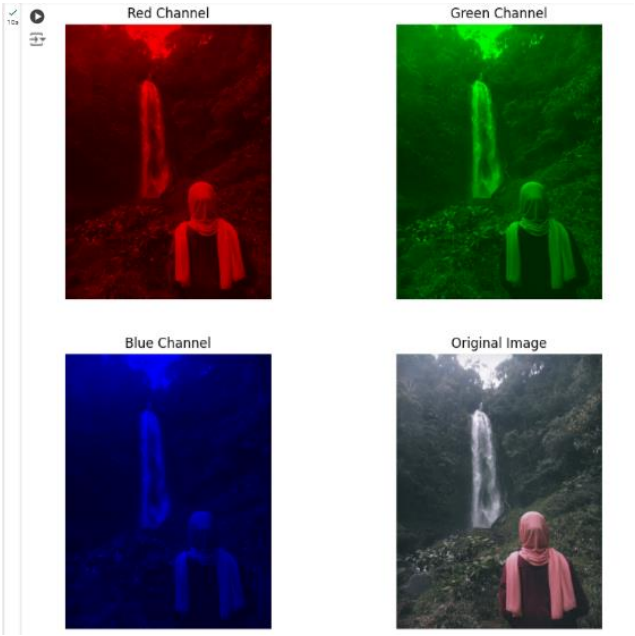
```
plt.subplot(2, 2, 2)
green_image = cv2.merge([zeros, G, zeros])  # G channel with R, B as 0
plt.imshow(green_image)
plt.title('Green Channel')
plt.axis('off')

# Blue channel
plt.subplot(2, 2, 3)
blue_image = cv2.merge([zeros, zeros, B])  # B channel with R, G as 0
plt.imshow(blue_image)
plt.title('Blue Channel')
plt.axis('off')

# Original image for comparison
plt.subplot(2, 2, 4)
plt.imshow(image_rgb)
plt.title('Original Image')
plt.axis('off')

plt.show()
```

2. Hasil Gambar



3. Hasil Analisis Kode Program

Untuk kode program pertama ini dibuat untuk mendekomposisi 3 warna RGB (Red, Green, Blue) secara terpisah dengan tujuan untuk menampilkan dan memahami bagaimana tiap kanal warna berkontribusi terhadap gambar asli, yang mana gambar digital yang ditampilkan diberi warna yang berbeda-beda diantara nya untuk gambar digital yang pertama diberi kanal warna merah sehingga untuk hasil gambar nya menjadi memiliki warna dengan dominan merah. Untuk Gambar yang kedua dan yang ketiga juga sama diberi kanal warna hijau atau biru sehingga hasil gambarnya dominan berwarna hijau atau biru. Pada gambar ke empat itu menampilkan gambar asli setelah di konversi ke RGB atau gambar asli itu dihasilkan dari campuran tiga kanal warna yaitu RGB.

Untuk menghasilkan gambar-gambar diatas itu dilakukan dengan cara memuat gambar dan mengkoversinya dari format BGR ke RGB, lalu memisahkan citra menjadi memiliki satu kanal diantara red, green dan blue, dengan menggunakan matriks nol untuk memisahkan satu kanal dari yang lainnya.

B. Analisis Kode Program Kedua

1. Kode Program

```
#Uniform Mapping dan Logarithmic Mapping
import cv2
import numpy as np
from matplotlib import pyplot as plt

# Step 1: Load an image (Replace the path with the actual image file path)
image_path = '/content/drive/MyDrive/Pengolahan Citra Digital/Melina.jpg'
# Ganti dengan path gambar Anda
image = cv2.imread(image_path)

# Step 2: Convert the image from BGR (OpenCV default) to RGB
image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

# Step 3: Split the image into its Red, Green, and Blue channels
R, G, B = cv2.split(image_rgb)

# Step 4: Create a zero matrix to help in visualization
zeros = np.zeros(image.shape[:2], dtype="uint8")

# Uniform Mapping (Normalize intensity between 0 and 255)
def uniform_mapping(channel):
    return cv2.normalize(channel, None, 0, 255, cv2.NORM_MINMAX)

# Logarithmic Mapping
def logarithmic_mapping(channel):
    c = 255 / np.log(1 + np.max(channel))
    log_mapped = c * (np.log(channel + 1))
    return np.array(log_mapped, dtype=np.uint8)

# Step 5: Apply Uniform and Logarithmic Mapping to each channel
R_uniform = uniform_mapping(R)
G_uniform = uniform_mapping(G)
B_uniform = uniform_mapping(B)

R_log = logarithmic_mapping(R)
G_log = logarithmic_mapping(G)
B_log = logarithmic_mapping(B)

# Step 6: Visualize the R, G, B channels separately and with
Uniform/Logarithmic mappings
plt.figure(figsize=(15, 15))

# Original Red channel
```

```
plt.subplot(3, 3, 1)
red_image = cv2.merge([R, zeros, zeros])  # R channel with G, B as 0
plt.imshow(red_image)
plt.title('Red Channel (Original)')
plt.axis('off')

# Uniform Red
plt.subplot(3, 3, 2)
uniform_red_image = cv2.merge([R_uniform, zeros, zeros])
plt.imshow(uniform_red_image)
plt.title('Red Channel (Uniform Mapping)')
plt.axis('off')

# Logarithmic Red
plt.subplot(3, 3, 3)
log_red_image = cv2.merge([R_log, zeros, zeros])
plt.imshow(log_red_image)
plt.title('Red Channel (Logarithmic Mapping)')
plt.axis('off')

# Original Green channel
plt.subplot(3, 3, 4)
green_image = cv2.merge([zeros, G, zeros])  # G channel with R, B as 0
plt.imshow(green_image)
plt.title('Green Channel (Original)')
plt.axis('off')

# Uniform Green
plt.subplot(3, 3, 5)
uniform_green_image = cv2.merge([zeros, G_uniform, zeros])
plt.imshow(uniform_green_image)
plt.title('Green Channel (Uniform Mapping)')
plt.axis('off')

# Logarithmic Green
plt.subplot(3, 3, 6)
log_green_image = cv2.merge([zeros, G_log, zeros])
plt.imshow(log_green_image)
plt.title('Green Channel (Logarithmic Mapping)')
plt.axis('off')

# Original Blue channel
plt.subplot(3, 3, 7)
blue_image = cv2.merge([zeros, zeros, B])  # B channel with R, G as 0
plt.imshow(blue_image)
plt.title('Blue Channel (Original)')
```

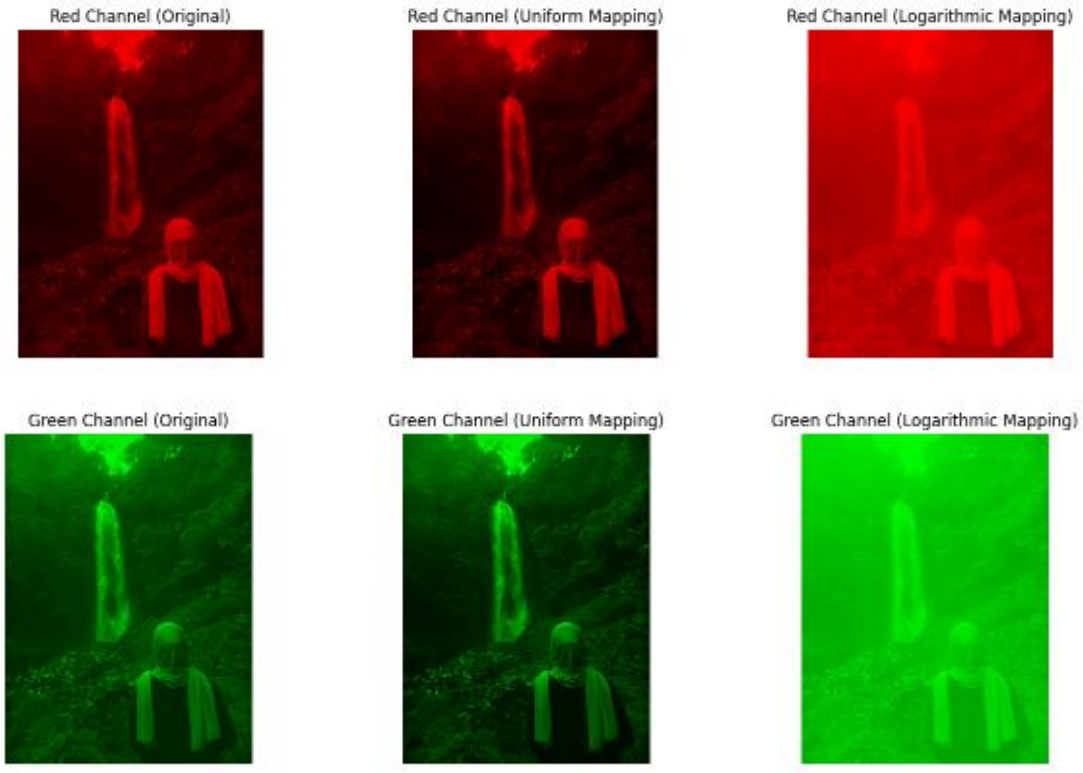
```
plt.axis('off')

# Uniform Blue
plt.subplot(3, 3, 8)
uniform_blue_image = cv2.merge([zeros, zeros, B_uniform])
plt.imshow(uniform_blue_image)
plt.title('Blue Channel (Uniform Mapping)')
plt.axis('off')

# Logarithmic Blue
plt.subplot(3, 3, 9)
log_blue_image = cv2.merge([zeros, zeros, B_log])
plt.imshow(log_blue_image)
plt.title('Blue Channel (Logarithmic Mapping)')
plt.axis('off')

plt.show()
```

2. Hasil Gambar



3. Hasil Analisis Kode Program

Untuk kode program kedua juga sama melakukan dekomposisi warna RGB namun ditambah dengan memberikan intensitas cahaya dengan dua metode yaitu metode Uniform mapping dan Logarithmic mapping dimana untuk uniform mapping dilakukan dengan cara menyebarkan intensitas piksel dengan merata ke seluruh rentang yang mungkin atau keseluruh nilai-nilai minimum dan maksimum yang bisa dimiliki oleh intensitas piksel dalam sebuah gambar, misalnya dengan nilai 0 sampai 255, yang mana jika semakin kecil nilainya atau mendekati 0 maka hasil intensitasnya akan terlihat lebih gelap, jika nilainya semakin besar atau mendekati 255 maka hasil intensitas pada gambarnya akan lebih terang. Untuk Uniform Mapping sering digunakan pada gambar yang memiliki kontras rendah, di mana sebagian besar piksel memiliki intensitas dalam rentang yang kecil.

Hasil dari gambar diatas, yang original menampilkan intensitas warna sesuai aslinya tanpa penyesuaian, sehingga area terang dan gelap mungkin kurang detail. Uniform Mapping menyebarkan intensitas piksel secara merata dari 0 hingga 255, meningkatkan kontras dan membuat detail lebih jelas di seluruh gambar. Di sisi lain, Logarithmic Mapping memperbesar intensitas rendah di area gelap, menonjolkan detail yang tidak terlihat pada gambar asli, sementara intensitas tinggi di area terang dikompresi untuk mencegah kelebihan cahaya. Hasilnya, Logarithmic Mapping memberikan keseimbangan yang baik antara area gelap dan terang.

C. Kesimpulan Hasil Analisis kedua Program

Kode pertama berfokus pada dekomposisi dasar gambar berwarna menjadi tiga kanal dan memvisualisasikan tiap kanal tanpa modifikasi intensitas, yang memberikan pandangan dasar bagaimana distribusi warna bekerja dalam gambar. Kode kedua melakukan analisis yang lebih dalam dengan menambahkan teknik pemetaan intensitas (Uniform dan Logarithmic) untuk meningkatkan detail atau menyesuaikan kontras. Teknik ini sangat berguna dalam pengolahan citra digital, terutama dalam konteks peningkatan kualitas gambar yang memiliki masalah dengan kontras atau detail dalam area gelap.