TUGAS 1
PENGELOLAAN CITRA DIGITAL

Untuk Memenuhi Pertemuan-4

“Mencoba Kode Program Yang Ada Pada PPT Pertemuan 4 Dengan Menggunakan Bahasa Pemrograman Bebas”

Disusun Oleh :

Agil Rahmat (2106037)

Informatika A

PROGRAM STUDI TEKNIK INFORMATIKA

INSTITUT TEKNOLOGI GARUT

2023

Penjelasan Kode Program

1. Alokasi Memori Untuk Matriks Citra

```
import numpy as np

# Ukuran gambar (misalnya, 640x480)
lebar = 640
tinggi = 480

# Mengalokasikan memori untuk matriks citra dengan tipe data
uint8 (8-bit per channel)
gambar = np.zeros((tinggi, lebar, 3), dtype=np.uint8)

# Anda dapat mengisi matriks citra dengan data atau warna sesuai
kebutuhan.

# Misalnya, mengisi seluruh matriks citra dengan warna merah
gambar[:, :, 0] = 255  # Channel merah diatur menjadi maksimum
(255)

# Menampilkan gambar
from matplotlib import pyplot as plt
plt.imshow(gambar)
plt.axis('off')  # Untuk menghilangkan sumbu x dan y
plt.show()
```

Output:

Penjelasan:

Kita dapat menggunakan library seperti NumPy untuk membuat dan mengelola matriks citra. Dalam contoh di atas, saya menggunakan NumPy untuk membuat matriks citra dengan ukuran tertentu dan menginisialisasinya dengan warna merah, tetapi kita dapat mengganti warna atau mengisi matriks sesuai kebutuhan. Setelah mengalokasikan matriks citra, kita dapat menggunakannya untuk manipulasi gambar, analisis citra, atau tujuan lainnya.

2. Dealokasi Memori Untuk Matriks Citra

```
import numpy as np

# Membuat matriks citra
gambar = np.zeros((480, 640, 3), dtype=np.uint8)

# Menghapus referensi ke matriks citra
del gambar

# Python akan merawat dealokasi memori jika tidak ada lagi
referensi ke matriks citra
```

Penjelasan:

Kita tidak perlu secara eksplisit melakukan dealokasi memori untuk variabel seperti matriks citra. Python mengelola alokasi dan dealokasi memori secara otomatis melalui mekanisme pengumpulan sampah (garbage collection). Ketika variabel tersebut tidak lagi digunakan, Python akan secara otomatis membersihkan memori yang digunakan oleh variabel tersebut.

Kita dapat menghapus referensi ke matriks citra menggunakan perintah del jika kita ingin melepaskan referensi tersebut, tetapi ini biasanya tidak diperlukan dalam kebanyakan kasus, dan Python akan merawat dealokasi memori dengan baik.

Namun, ingatlah bahwa penggunaan del umumnya tidak diperlukan dalam kode Python sehari-hari, kecuali dalam situasi-situasi khusus. Python akan secara otomatis mengelola alokasi dan dealokasi memori dengan baik, dan perluasan memori biasanya dilakukan secara otomatis oleh sistem.

3. Fungsi Alokasi Secara Umum

```
def alokasi_memori_untuk_array(ukuran):
    return [0] * ukuran

# Menggunakan fungsi alokasi untuk membuat array
ukuran_array = 10
data = alokasi_memori_untuk_array(ukuran_array)

# Melakukan sesuatu dengan array yang dialokasikan
for i in range(ukuran_array):
    data[i] = i

# Menampilkan array
print(data)
```

Output:

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

Penjelasan:

Fungsi yang digunakan untuk mengalokasikan sumber daya, seperti memori atau penyimpanan, yang diperlukan untuk menyimpan data atau objek tertentu. Dalam bahasa Python, kita tidak perlu secara eksplisit menulis fungsi alokasi memori, karena alokasi memori untuk objek biasanya dilakukan oleh interpreter Python secara otomatis. Namun, kita dapat membuat fungsi yang mengalokasikan sumber daya tambahan sesuai kebutuhan.

Dalam contoh di atas, alokasi_memori_untuk_array adalah fungsi yang mengalokasikan array dengan panjang tertentu dan mengembalikan array tersebut. Kemudian, kita mengisi array dengan data sesuai kebutuhan.

Fungsi alokasi bisa menjadi lebih kompleks tergantung pada kebutuhan proyek. Misalnya, kita dapat membuat fungsi yang mengalokasikan memori dinamis untuk gambar, matriks, atau objek lainnya. Namun, dalam Python, kita sering tidak perlu khawatir tentang alokasi memori secara eksplisit, kecuali dalam kasus-kasus khusus.

4. Menampilkan Citra Ke Layar

```
import matplotlib.pyplot as plt
import matplotlib.image as mpimg


# Load the image
img = mpimg.imread('Lena.jpeg')


# Display the image
plt.imshow(img)
plt.axis('off')  # Untuk menghilangkan sumbu x dan y
plt.show()
```

Output:

Penjelasan:

Kita dapat menggunakan library Python seperti Matplotlib atau OpenCV. Di atas ini adalah contoh kode program untuk menampilkan citra menggunakan Matplotlib.

Pastikan kita mengganti 'nama_file_gambar.png' dengan nama file gambar yang ingin tampilkan.

Jika kita ingin menggunakan OpenCV untuk menampilkan gambar, berikut contohnya seperti dibawah:

```
from IPython.display import Image, display


# Menampilkan gambar JPEG
display(Image(filename="Lena.jpeg"))
```

Output:

Penjelasan:

Kembali, pastikan untuk mengganti 'nama_file_gambar.png' dengan nama file gambar yang ingin ditampilkan. Juga, perlu diingat bahwa dalam kasus ini, kita harus menggunakan cv2_imshow yang merupakan fungsi yang disediakan oleh Google Colab untuk menampilkan gambar dengan OpenCV.

5. Menampilkan Citra Grayscale 8-Bit

```
import cv2
from google.colab.patches import cv2_imshow

# Membaca gambar grayscale dari file
gambar = cv2.imread('Lena.jpeg', cv2.IMREAD_GRAYSCALE)

# Menampilkan gambar grayscale
cv2_imshow(gambar)
```

Output:

Penjelasan:

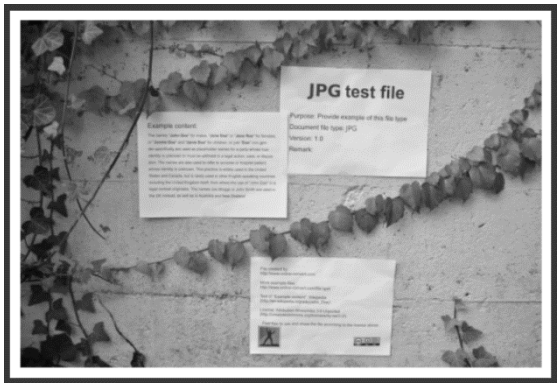
Kita dapat menggunakan library seperti OpenCV atau Matplotlib. Di atas ini adalah contoh penggunaan OpenCV

Pastikan kita mengganti 'nama_file_gambar.jpg' dengan nama file gambar grayscale yang ingin Anda tampilkan.

KIta juga bisa menggunakan Matplotlib jika lebih suka, seperti program dibawahnya:

```
import matplotlib.pyplot as plt


# Membaca gambar grayscale dari file
gambar = cv2.imread('example.jpg', cv2.IMREAD_GRAYSCALE)


# Menampilkan gambar grayscale
plt.imshow(gambar, cmap='gray')
plt.axis('off')
plt.show()
```

Output:



Penjelasan:

Dalam contoh ini, kami menggunakan OpenCV untuk membaca citra grayscale dan menampilkannya dengan fungsi cv2_imshow, atau menggunakan Matplotlib untuk menampilkannya dengan plt.imshow. Pastikan kita memiliki gambar grayscale yang sesuai dengan format yang kita pilih (OpenCV atau Matplotlib) dan bahwa kita menggantinya dengan nama file gambar yang sesuai.

6. Perubahan Jika Citra Yang Ditampilkan Adalah Berwarna

```
import cv2
from google.colab.patches import cv2_imshow


# Membaca gambar berwarna dari file
```

```
gambar = cv2.imread('Lena.jpeg')

# Mengubah gambar menjadi citra grayscale
gambar_grayscale = cv2.cvtColor(gambar, cv2.COLOR_BGR2GRAY)

# Menampilkan citra grayscale
cv2_imshow(gambar_grayscale)
```

Output:

Penjelasan:

Jika kita ingin mengubah citra berwarna (RGB) menjadi citra grayscale. Kita dapat menggunakan OpenCV atau Matplotlib. Di atas ini adalah contoh menggunakan OpenCV:

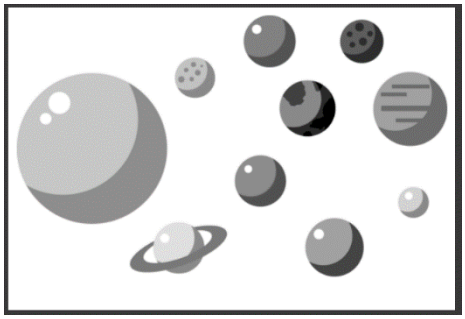
Dalam contoh ini, kita membaca citra berwarna dari file, mengubahnya menjadi citra grayscale menggunakan cv2.cvtColor, dan menampilkannya dengan cv2_imshow.

Jika kita lebih suka menggunakan Matplotlib, kita bisa melakukan hal berikut, seperti yang dibawah:

```
import cv2
import matplotlib.pyplot as plt

# Membaca gambar berwarna dari file
```

```
gambar = cv2.imread('wing.png')

# Mengubah gambar menjadi citra grayscale
gambar_grayscale = cv2.cvtColor(gambar, cv2.COLOR_BGR2GRAY)

# Menampilkan citra grayscale
plt.imshow(gambar_grayscale, cmap='gray')
plt.axis('off')
plt.show()
```

Output:



Penjelasan:

Pastikan kita mengganti 'nama_file_gambar.jpg' dengan nama file gambar berwarna yang ingin kita konversi menjadi citra grayscale dan tampilkan.

7. Membaca File Mentah

```
import cv2

# Membaca citra dari file mentah
gambar_mentah = cv2.imread('LenaR.raw',
cv2.IMREAD_UNCHANGED)

# Menampilkan citra yang dibaca
from google.colab.patches import cv2_imshow
cv2_imshow(gambar_mentah)
```

Output:

Penjelasan:

Untuk membaca citra dari file citra mentah (raw image file), kita perlu menggunakan library seperti OpenCV atau PIL (Python Imaging Library).

Pada kode di atas, kita menggunakan cv2.imread dengan opsi cv2.IMREAD_UNCHANGED untuk membaca citra mentah. Pastikan kita mengganti 'nama_file_mentah.raw' dengan nama file citra mentah yang ingin dibaca.

Namun, perlu diingat bahwa format citra mentah bervariasi tergantung pada kamera atau perangkat yang digunakan untuk menghasilkannya. Kita perlu tahu format file citra mentah yang dimiliki dan pastikan bahwa OpenCV mendukungnya. Jika formatnya tidak didukung secara langsung oleh OpenCV, kita mungkin perlu melakukan konversi atau pengolahan tambahan sesuai dengan format file mentah yang digunakan.

8. Menyimpan Citra Ke File Mentah

```
import numpy as np


# Membaca citra dari file yang akan disimpan
gambar = cv2.imread('example.jpg', cv2.IMREAD_UNCHANGED)


# Simpan data citra ke dalam file mentah
with open('EX.raw', 'wb') as file:
```

```
        file.write(gambar.tobytes())
```

Penjelasan:

Untuk menyimpan citra ke file mentah (raw image file), kita perlu melakukan beberapa langkah tambahan karena OpenCV tidak memiliki dukungan langsung untuk format mentah. Kita perlu mengelola data citra secara manual dan menyimpannya ke dalam format yang sesuai. Diatas contoh cara menyimpan citra ke file mentah dengan bantuan NumPy.

Dalam kode di atas, kita membaca citra dari file gambar dengan menggunakan cv2.imread dengan opsi cv2.IMREAD_UNCHANGED untuk memastikan citra diambil dalam format asli. Kemudian, kita menggunakan tobytes() untuk mengonversi data citra menjadi byte array dan menyimpannya dalam file mentah menggunakan open dengan mode 'wb' (write binary).

Pastikan untuk mengganti 'nama_file_gambar.jpg' dengan nama file gambar yang ingin disimpan dalam format mentah, dan sesuaikan format data mentah jika diperlukan sesuai dengan kebutuhan.