Segmentasi Gambar Berdasarkan Warna Dominan Menggunakan K-Means Clustering

Nama : Rezal Suryadi Putra

Kelas : AIG-B

NIM : 2250081107

1. Latar Belakang

Dalam dunia pengolahan citra digital, sering kali kita ingin menyederhanakan gambar agar lebih mudah dianalisis atau dikompresi. Salah satu pendekatan yang umum digunakan adalah teknik segmentasi berdasarkan warna. K-Means Clustering dapat digunakan untuk mengelompokkan piksel berdasarkan kemiripan warna, sehingga menghasilkan representasi gambar yang lebih sederhana namun tetap informatif.

2. Tujuan

Mengimplementasikan algoritma K-Means untuk melakukan segmentasi warna pada gambar digital secara otomatis.

3. Metode

Metode yang digunakan dalam proyek ini meliputi:
- Membaca dan memuat gambar dari Google Drive
- Mengubah piksel gambar menjadi vektor RGB
- Melakukan clustering menggunakan K-Means
- Menampilkan hasil segmentasi warna berdasarkan jumlah klaster (warna dominan)

4. Dataset

Gambar diambil dari Google Drive pengguna secara manual.

5. Implementasi dan Analisis Kode

[1] Mount Google Drive

Kode ini digunakan untuk menghubungkan Google Colab dengan akun Google Drive pengguna sehingga gambar bisa diakses langsung dari penyimpanan Drive.

```
from google.colab import drive
drive.mount('/content/drive')
```

[2] Import Library dan Load Gambar

Berbagai library seperti numpy, matplotlib, PIL, dan scikit-learn digunakan untuk pemrosesan gambar dan clustering. Gambar dibaca dan diubah menjadi array RGB yang dinormalisasi antara 0 dan 1. Data kemudian diubah ke bentuk vektor.

```
import numpy as np
import matplotlib.pyplot as plt
from PIL import Image
from sklearn.cluster import KMeans
from sklearn.utils import shuffle
import ipywidgets as widgets
from IPython.display import display

# Load gambar
img_path =
'/content/drive/MyDrive/folder_gambar/nama_file.jpg
'
original_image =
Image.open(img_path).convert('RGB')
img = np.array(original_image, dtype=np.float64) /
255
w, h, d = img.shape
image_array = np.reshape(img, (w * h, d))
image_array_sample = shuffle(image_array,
random_state=42)[:1000]
```

[3] Fungsi Clustering dan Visualisasi

Fungsi `cluster_and_display` menjalankan K-Means pada sampel data piksel. Setelah clustering, hasil dikembalikan dalam bentuk gambar yang direkonstruksi dari warna-warna klaster. Gambar asli dan hasil clustering ditampilkan berdampingan.

```
def cluster_and_display(n_colors):
    kmeans = KMeans(n_clusters=n_colors,
random_state=42).fit(image_array_sample)
    labels = kmeans.predict(image_array)
    clustered_img = kmeans.cluster_centers_[labels]
    clustered_img = np.reshape(clustered_img, (w,
h, d))

    plt.figure(figsize=(10, 5))
```

```
plt.subplot(1, 2, 1)
plt.imshow(original_image)
plt.title("Gambar Asli")
plt.axis('off')

plt.subplot(1, 2, 2)
plt.imshow(clustered_img)
plt.title(f"Hasil K-Means (n={n_colors})")
plt.axis('off')

plt.show()
```

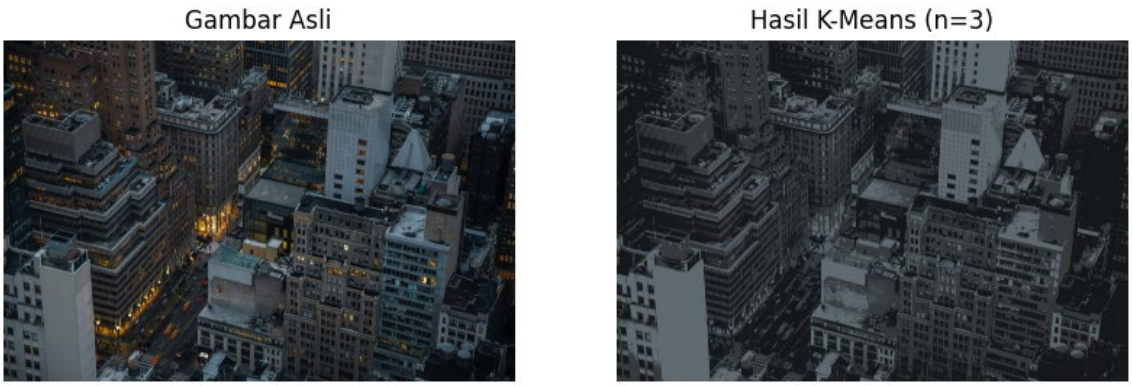
[4] Slider Interaktif

Kode ini membuat slider interaktif yang memungkinkan pengguna memilih jumlah klaster warna antara 2 hingga 15. Slider ini bekerja dengan memanggil fungsi `cluster_and_display` setiap kali nilainya diubah.

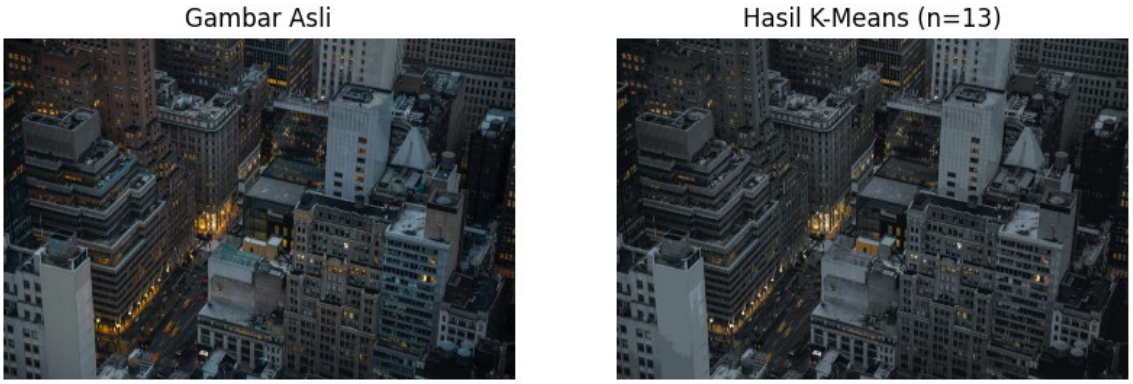
```
slider = widgets.IntSlider(value=5, min=2, max=15,
step=1, description='Clusters:')
widgets.interact(cluster_and_display,
n_colors=slider)
```

6. Hasil

Setelah kode dijalankan, pengguna dapat melihat hasil clustering warna pada gambar dengan jumlah klaster yang bervariasi. Semakin sedikit jumlah klaster, semakin sederhana representasi warna gambar.



Gambar 1 K-3



Gambar 2 K-13

7. Kesimpulan

K-Means Clustering dapat digunakan untuk menyederhanakan gambar dengan cara mengelompokkan warna dominan. Dengan mengurangi jumlah warna, hasil segmentasi tetap mempertahankan struktur gambar asli namun dalam bentuk yang lebih ringkas.