LAPORAN TUGAS

IMAGE PROCESSING

JUDUL:

Tugas Image Enhancement

OLEH KELOMPOK 5

Muhammad Zaki Al Hafiz	:	2211533004
Zaky Adil Hakim	:	2211533007
Muhammad Zhafarul Maahiy	:	2211533009
Humayra Fahreri	:	2211533019

DEPARTEMEN INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

UNIVERSITAS ANDALAS

2025

Image Enhancement

1. Pengantar Image Enhancement

Image Enhancement atau peningkatan kualitas citra merupakan proses dalam pengolahan citra digital yang bertujuan untuk memperbaiki kualitas citra dan memperoleh kualitas citra yang lebih baik sehingga dapat digunakan untuk interpretasi lebih lanjut. Tidak hanya itu, image enhancement ini juga merupakan salah satu proses awal dalam pengolahan citra atau yang disebut dengan *preprocessing*.

Image Enhancement sangat diperlukan dalam pengolahan citra dikarenakan:

- Citra mengandung noise
- Citra terlalu terang atau citra terlalu gelap, kurang tajam atau blur
- Cacat ketika melakukan proses akuisisi citra
    - lensa: object blurring atau background blurring
    - objek bergerak kamera bergerak: motion blurring
- Distorsi geometric akibat sudut pengambilan gambar atau karakteristik lensa

Noisy image

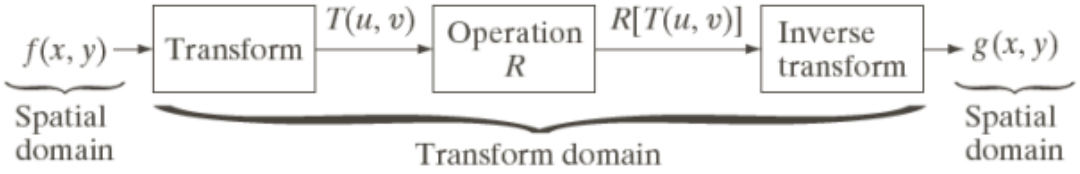
Citra dengan kontras terlalu gelap

Motion blur

Image enhancement dibagi menjadi dua kategori berdasarkan domain operasinya, meliputi:

- Ranah Spasial (Spatial Domain): memanipulasi langsung nilai piksel dalam citra.

- Ranah Frekuensi; dilakukan dengan mengubah citra terlebih dahulu dari spasial ke frekuensi. Setelah selesai mengubah citra, dilanjutkan dengan manipulasi nilai frekuensi.



2. Metode Image Enhancement dalam Ranah Spasial

Pada ranah spasial, setiap piksel dalam citra diproses menggunakan operator T, dengan persamaan:

$$g(x,y)=T[f(x,y)]$$

di mana:

- f(x,y) = citra input,
- g(x,y) = citra output,
- T = operator transformasi.

3. Proses dalam Image Enhancement

a. Image Brightening

Dilakukan dengan menambahkan atau mengalikan setiap piksel dengan konstanta:

$$s = r + b$$

Di mana:

- r = intensitas piksel input
- b = konstanta (positif untuk meningkatkan kecerahan, negatif untuk mengurangi cahaya)

b. Citra Negatif

Untuk mengubah citra menjadi negative, digunakan persamaan berikut:

$$s = (L - 1) - r$$

L = 256 untuk citra grayscale 8-bit

Digunakan untuk menyoroti detail pada citra dengan dominasi warna hitam, misalnya foto sinar X

c. Transformasi Logaritma

Menggunakan fungsi:

$$s = c \log(1+r)$$

transformasi logaritma ini mampu meningkatkan detail pada area gelap dan mengurangi intensitas pada area terang.

d. Transformasi Pangkat

Menggunakan rumus:

$$s = cr^{\gamma}$$

di mana c dan γ adalah konstanta positif.

- Jika $\gamma < 1$: Meningkatkan detail pada area gelap.
- Jika $\gamma > 1$: Meningkatkan detail pada area terang.

e. Contrast Stretching

Digunakan untuk meningkatkan rentang nilai keabuan citra dengan menerapkan fungsi linier:

$$s = \frac{255}{r_{max} - r_{min}} . (r - r_{min})$$

$r_{max} - r_{min}$ merupakan batas bawah dan atas dari rentang nilai intensitas

Metode lain dalam contrast stretching:

- Thresholding: Mengubah nilai piksel di bawah ambang batas menjadi hitam dan di atas ambang batas menjadi putih.
- Adaptive Contrast Enhancement: Digunakan untuk citra dengan kontras rendah akibat pencahayaan buruk.

f. Gray-Level Slicing

Menonjolkan rentang keabuan tertentu dalam citra dengan dua pendekatan:

- Discard background: Membuat area tertentu lebih terang, sementara area lain lebih gelap.
- Preserve background: Hanya meningkatkan kecerahan bagian tertentu tanpa mengubah bagian lainnya.

g. Bit-Plane Slicing

Mengekstrak informasi dari setiap bit dalam nilai intensitas piksel.

Struktur 8-bit dalam citra grayscale:

$$b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0$$

- Bit MSB (Most Significant Bits) mengandung informasi utama citra.
- Bit LSB (Least Significant Bits) mengandung detail kecil atau noise.

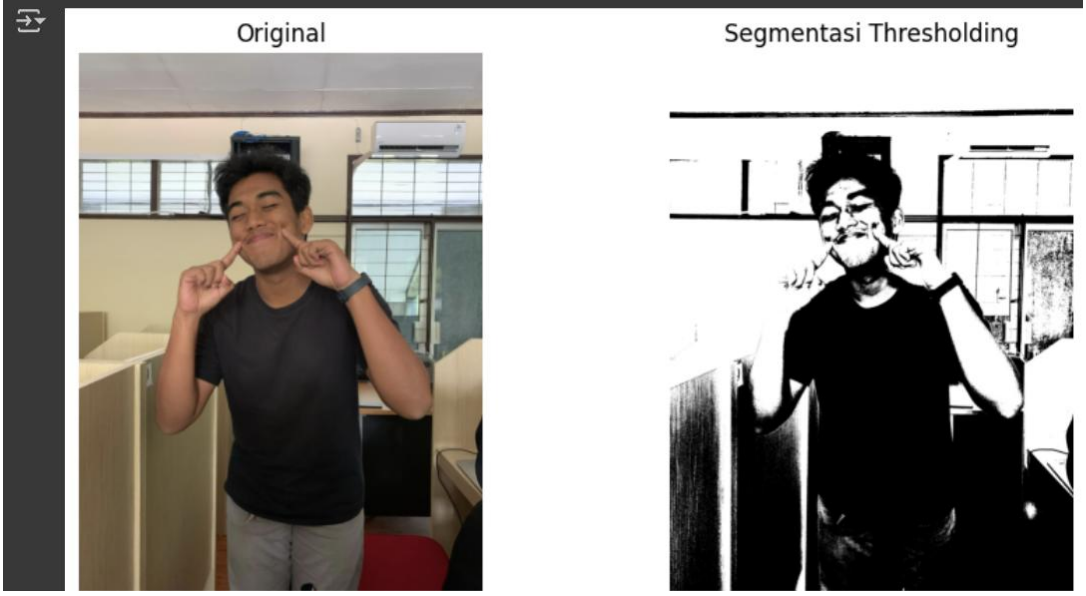
4. Implementasi Program Menggunakan Python

a. Import library PIL (Pillow) untuk manipulasi gambar, numpy untuk operasi numerik, dan matplotlib.pyplot untuk visualisasi. Fungsi load_image(path) digunakan untuk memuat gambar dari path drector.

```
from PIL import Image, ImageOps
import numpy as np
import matplotlib.pyplot as plt

# Load image
def load_image(path):
    return Image.open(path)
```

b. Pada fungsi img.convert("L") gambar dikonversi ke dalam bentuk grayscale. Mode "L" dalam Pillow berarti Luminance (hitam-putih) dengan skala 0-255, di mana “0” berarti warna hitam gelap dan “255” berarti warna putih. Pada “grayscale.point(lambda p: 255 if p > threshold else 0)” Menggunakan fungsi point() untuk menerapkan thresholding ke setiap piksel. Dengan cara kerja jika nilai piksel lebih dari threshold (misalnya 100), piksel diubah menjadi 255 (putih), dan jika nilai piksel kurang atau sama dengan threshold, piksel diubah menjadi 0 (hitam). Fungsi memberikan nilai output berupa gambar hitam putih. Fungsi “show_comparison” digunakan untuk menampilkan perbandingan gambar sebelum dilakukannya grayscale pada gambar. Lalu kedua fungsi dipanggil untuk dijalankan.

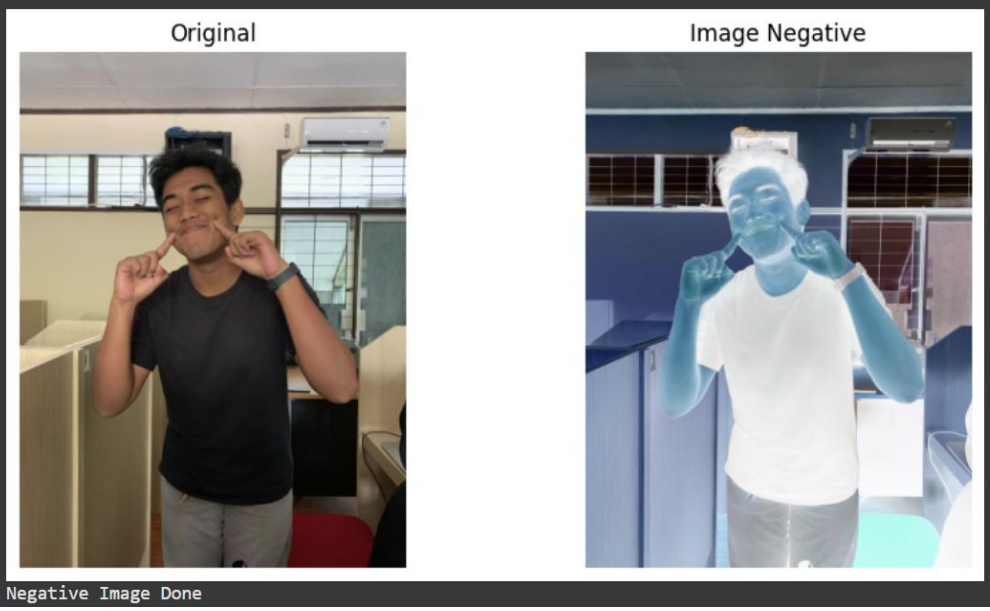
```python
# Segmentasi menggunakan image thresholding
def image_threshold(img, threshold=100):
    grayscale = img.convert("L")
    binary = grayscale.point(lambda p: 255 if p > threshold else 0)
    return binary

# Fungsi untuk menampilkan perbandingan gambar
def show_comparison(original, processed, title):
    plt.figure(figsize=(10, 5))
    plt.subplot(1, 2, 1)
    plt.imshow(original)
    plt.title("Original")
    plt.axis('off')

    plt.subplot(1, 2, 2)
    plt.imshow(processed, cmap="gray")
    plt.title(title)
    plt.axis('off')
    plt.show()

# Penggunaan Fitur
if __name__ == "__main__":
    image_path = "AsepSiImutDariPYK.jpg"
    img = load_image(image_path)

   # Segmentasi Thresholding
    threshold_img = image_threshold(img)
    threshold_img.save("Asep_Segmented.jpg")
    show_comparison(img, threshold_img, "Segmentasi Thresholding")
    print("Segmentation Thresholding done")
```



c. Pembuatan fungsi image_negative untuk melakukan proses image negative pada gambar. Pada method “img.convert("RGB")” digunakan untuk mengonversi gambar ke mode **RGB** (Red-Green-Blue), Ini memastikan bahwa operasi inversi dapat diterapkan secara langsung pada gambar berwarna. Pada

“ImageOps.invert(img.convert("RGB"))” ImageOps.invert() akan membalikkan warna setiap piksel dari gambar yang dimasukkan sehingga piksel dengan nilai terang menjadi gelap dan sebaliknya. Fungsi “show_comparison” digunakan untuk menampilkan perbandingan gambar sebelum dilakukannya proses image negative pada gambar. Lalu kedua fungsi dipanggil untuk dijalankan.

```python
# Mengubah citra menjadi image negative
def image_negative(img):
    return ImageOps.invert(img.convert("RGB"))

# Fungsi untuk menampilkan perbandingan gambar
def show_comparison(original, processed, title):
    plt.figure(figsize=(10, 5))
    plt.subplot(1, 2, 1)
    plt.imshow(original)
    plt.title("Original")
    plt.axis('off')

    plt.subplot(1, 2, 2)
    plt.imshow(processed, cmap="gray")
    plt.title(title)
    plt.axis('off')
    plt.show()

# Penggunaan Fitur
if __name__ == "__main__":
    image_path = "AsepSiImutDariPYK.jpg"
    img = load_image(image_path)

# Image Negative
    negative_img = image_negative(img)
    negative_img.save("Asep_Negatif.jpg")
    show_comparison(img, negative_img, "Image Negative")
    print("Negative Image Done")
```

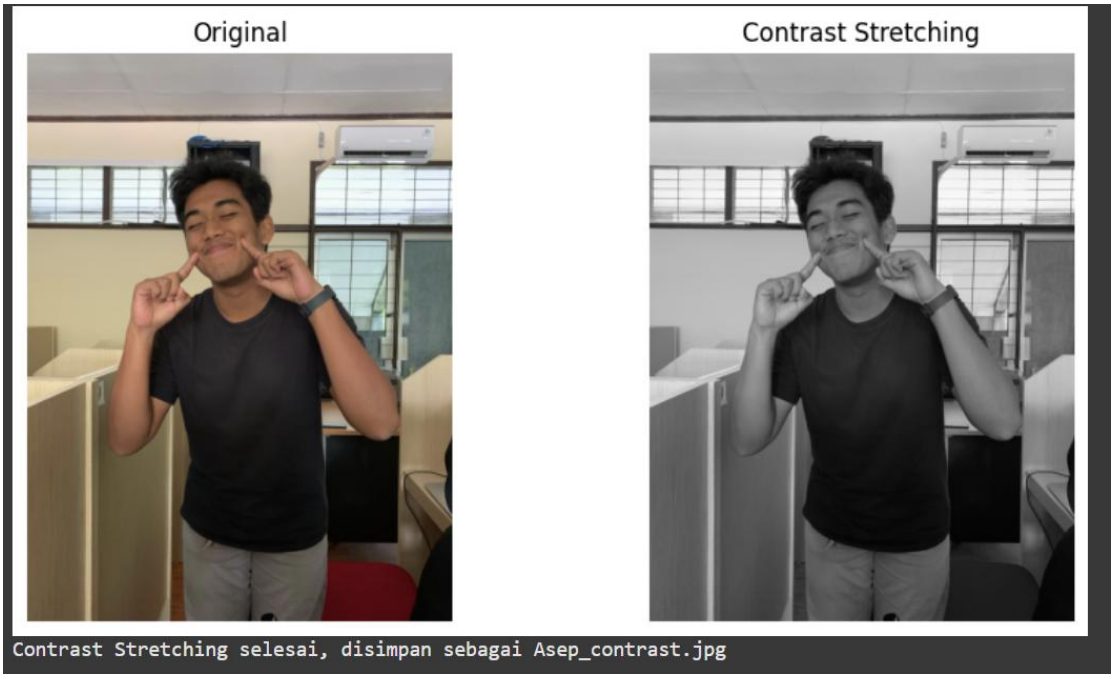



```
Negative Image Done
```

d. Contrast Stretching

Pertama mengubah gambar menjadi **grayscale** (hitam-putih) agar hanya memiliki satu kanal intensitas menggunkan method “img.convert("L")”. Gambar yang telah menjadi grayscale di konversi kedalam bentuk array NumPy untuk memudahkan operasi matematika menggunakan method “np.array(img.convert("L"))”. Lalu mencari nilai intensitas piksel terendah dalam gambar menggunakan method ” min_val = np.min(img_array)”. Lalu mencari nilai intensitas piksel tertinggi dalam gambar menggunakan method ” max_val = np.max(img_array)”. Selanjutnya mengubah rentang nilai intensitas dari [min_val, max_val] menjadi [0, 255] dan mengonversi ke uint agar sesuai dengan format citra dengan sintaks “stretched = ((img_array - min_val) / (max_val - min_val) * 255).astype(np.uint8)”. Lalu mengonversi array NumPy menjadi gambar untuk ditampilkan atau disimpan. Fungsi “show_comparison” digunakan untuk menampilkan perbandingan gambar sebelum dilakukannya proses image negative pada gambar. Lalu kedua fungsi dipanggil untuk dijalankan.

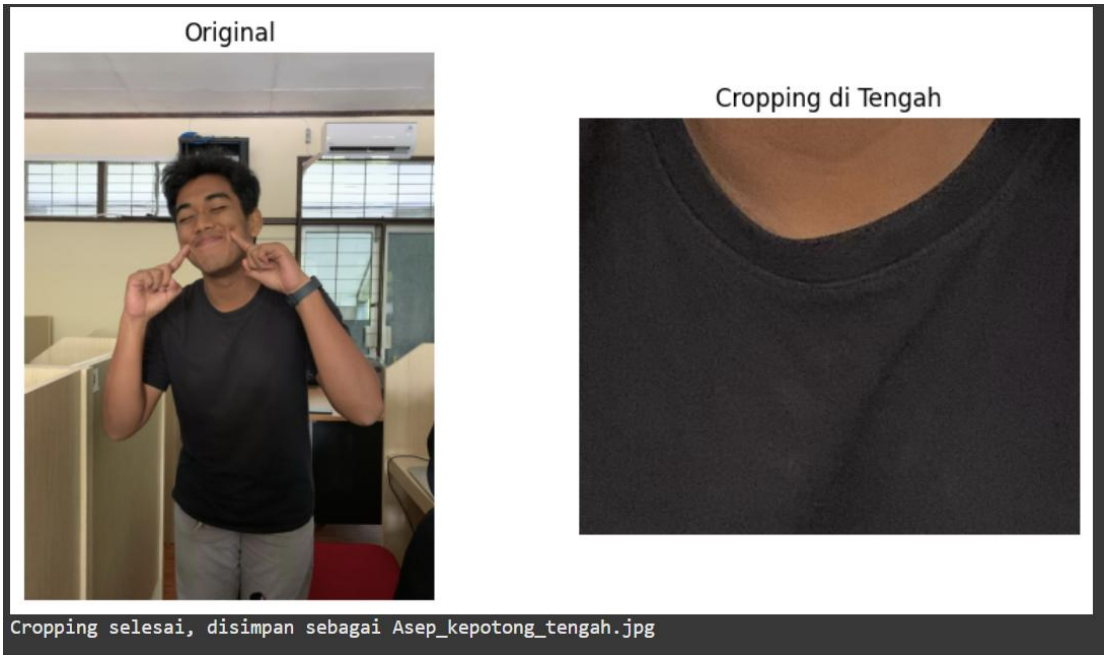
```python
# Meningkatkan kontras citra (contrast stretching)
def contrast_stretching(img):
    img_array = np.array(img.convert("L"))
    min_val = np.min(img_array)
    max_val = np.max(img_array)
    stretched = ((img_array - min_val) / (max_val - min_val) * 255).astype(np.uint8)
    return Image.fromarray(stretched)

# Fungsi untuk menampilkan perbandingan gambar
def show_comparison(original, processed, title):
    plt.figure(figsize=(10, 5))
    plt.subplot(1, 2, 1)
    plt.imshow(original)
    plt.title("Original")
    plt.axis('off')

    plt.subplot(1, 2, 2)
    plt.imshow(processed, cmap="gray")
    plt.title(title)
    plt.axis('off')
    plt.show()

# Penggunaan Fitur
if __name__ == "__main__":
    image_path = "AsepSiImutDariPYK.jpg"
    img = load_image(image_path)

 # Contrast Stretching
    contrast_img = contrast_stretching(img)
    contrast_img.save("Asep_Contrast.jpg")
    show_comparison(img, contrast_img, "Contrast Stretching")
    print("Contrast Stretching selesai, disimpan sebagai Asep_contrast.jpg")
```



e. Image Substraction

Pada bagian image substraction gambar akan dipotong dan diambil bagian tengahnya saja. Pertama membuat fungsi untuk melakukan cropping pada gambar dengan nama “image_crop_center”. Fungsi akan akan mengambil lebar dan tinggi gambar terlebih dahulu dengan sintaks “width, height = img.size”. selanjutnya menghitung jumlah piksel titik tengah ke kiri, atas, kanan dan bawah pada gambar dan menentukan titik potong pada gambar dengan sintaks “left = (width - crop_size[0]) // 2”(sumbu x) , “top = (height - crop_size[1]) // 2”(sumbu y), “right = left + crop_size[0]”(menentukan area potong x), “bottom = top + crop_size[1]”(menentukan area potong y). Lalu gambar akan dipotong dengan kordinat yang telah ditentukan dengan sintaks “return img.crop((left, top, right, bottom))”. Fungsi “show_comparison” digunakan untuk menampilkan perbandingan gambar sebelum dilakukannya proses image negative pada gambar. Lalu kedua fungsi dipanggil untuk dijalankan.

```python
from PIL import Image
import matplotlib.pyplot as plt

# Fungsi untuk memuat gambar
def load_image(image_path):
    return Image.open(image_path)

# Cropping citra di tengah
def image_crop_center(img, crop_size):
    width, height = img.size
    left = (width - crop_size[0]) // 2
    top = (height - crop_size[1]) // 2
    right = left + crop_size[0]
    bottom = top + crop_size[1]
    return img.crop((left, top, right, bottom))

# Fungsi untuk menampilkan perbandingan gambar
def show_comparison(original, processed, title):
    plt.figure(figsize=(10, 5))
    plt.subplot(1, 2, 1)
    plt.imshow(original)
    plt.title("Original")
    plt.axis('off')

    plt.subplot(1, 2, 2)
    plt.imshow(processed)
    plt.title(title)
    plt.axis('off')
    plt.show()

# Penggunaan Fitur
if __name__ == "__main__":
    image_path = "AsepSiImutDariPYK.jpg"
    img = load_image(image_path)

    # Ukuran cropping (lebar, tinggi)
    crop_size = (600, 500)  # Contoh: potong gambar menjadi 200x200 piksel


    # Cropping di tengah
    cropped_img = image_crop_center(img, crop_size)
    cropped_img.save("Asep_kepotong.jpg")
    show_comparison(img, cropped_img, "Cropping di Tengah")
    print("Cropping selesai, disimpan sebagai Asep_kepotong.jpg")
```



f. Normalisasi Histogram

Langkah pertama membuat fungsi yang akan digunakan untuk melakukan normalisasi histogram pada gambar dengan nama “histogram_normalization”. Pada fungsi gambar akan diubah ke warna abu-abu atau *grayscale* dengna sintaks “img_array = np.array(img.convert("L"))” karena normalisasi histogram biasanya digunakan pada gambar dengan warna monokrom. Gambar yang sudah berbentuk grayscale akan dihitung intensitasnya menggunakan sintaks “hist, bins = np.histogram(img_array.flatten(), 256, [0, 256])” dan dikonversi ke dalam bentuk array 1 dimensi untuk memudahkan perhitungan histogram dengan sintaks “img_array.flatten()”. Lalu pada gambar akan dihitung *cumulative distribution function* (CDF) untuk meratakan distribusi intensitas pada piksel gambar dengan menggunakan sintaks “cdf = hist.cumsum()”. CDF yang telah didapatkan akan dinormalisasikan agar rentang nilainya berada pada 0-255 dengan sintaks “cdf_normalized = (cdf - cdf.min()) * 255 / (cdf.max() - cdf.min())”. Selanjutnya melakukan transformasi intensitas piksel menggunakan interpolasi linier berdasarkan CDF yang telah dinormalisasikan dengan sintaks “img_normalized = np.interp(img_array.flatten(), bins[:-1], cdf_normalized)”. Selanjutnya mengubah array 1 dimensi ke dalam 2 dimensi dan dikonversi kembali kedalam bentuk gambar dengan sintaks “return Image.fromarray(img_normalized.reshape(img_array.shape).astype(np.uint8))”. Fungsi “show_comparison” digunakan untuk menampilkan perbandingan gambar sebelum dilakukannya proses image negative pada gambar. Lalu kedua fungsi dipanggil untuk dijalankan.

```python
# Normalisasi histogram dari citra
def histogram_normalization(img):
    img_array = np.array(img.convert("L"))
    hist, bins = np.histogram(img_array.flatten(), 256, [0, 256])
    cdf = hist.cumsum()
    cdf_normalized = (cdf - cdf.min()) * 255 / (cdf.max() - cdf.min())
    img_normalized = np.interp(img_array.flatten(), bins[:-1], cdf_normalized)
    return Image.fromarray(img_normalized.reshape(img_array.shape).astype(np.uint8))

# Fungsi untuk menampilkan perbandingan gambar
def show_comparison(original, processed, title):
    plt.figure(figsize=(10, 5))
    plt.subplot(1, 2, 1)
    plt.imshow(original)
    plt.title("Original")
    plt.axis('off')

    plt.subplot(1, 2, 2)
    plt.imshow(processed, cmap="gray")
    plt.title(title)
    plt.axis('off')
    plt.show()

# Penggunaan Fitur
if __name__ == "__main__":
    image_path = "AsepSiImutDariPYK.jpg"
    img = load_image(image_path)

  # Histogram Normalization
    hist_norm_img = histogram_normalization(img)
    hist_norm_img.save("Asep_Histogram.jpg")
    show_comparison(img, hist_norm_img, "Histogram Normalization")
    print("Histogram Normalization selesai, disimpan sebagai Asep_Histogram.jpg")
```