**PRAKTIKUM 9 EDGE DETECTION**

Nama : Muhammad Baehaqi

NIM : 1207070134

**Latihan 1 Filter Sobel**

**Source Code :**

import cv2

import numpy as np

from matplotlib import pyplot as plt

from skimage import data

image = cv2.imread("songhayoon.jpg",0)

image\_sobelx = cv2.Sobel(image, cv2.CV\_8U, 1, 0, ksize = 5)

image\_sobely = cv2.Sobel(image, cv2.CV\_8U, 0, 1, ksize = 5)

image\_sobel = image\_sobelx + image\_sobely

fig, axes = plt.subplots (4, 2, figsize = (20, 20))

ax = axes.ravel ()

ax[0].imshow(image, cmap = 'gray')

ax[0].set\_title ("Citra Input")

ax[1].hist (image.ravel(), bins = 256)

ax[1].set\_title ("Histogram Citra Input")

ax[2].imshow(image\_sobelx, cmap = 'gray')

ax[2].set\_title ("Citra Input")

ax[3].hist (image\_sobelx.ravel(), bins = 256)

ax[3].set\_title ("Histogram Citra Output")

ax[4].imshow(image\_sobely, cmap = 'gray')

ax[4].set\_title ("Citra Input")

ax[5].hist (image\_sobely.ravel(), bins = 256)

ax[5].set\_title ("Histogram Citra Output")

ax[6].imshow(image\_sobel, cmap = 'gray')

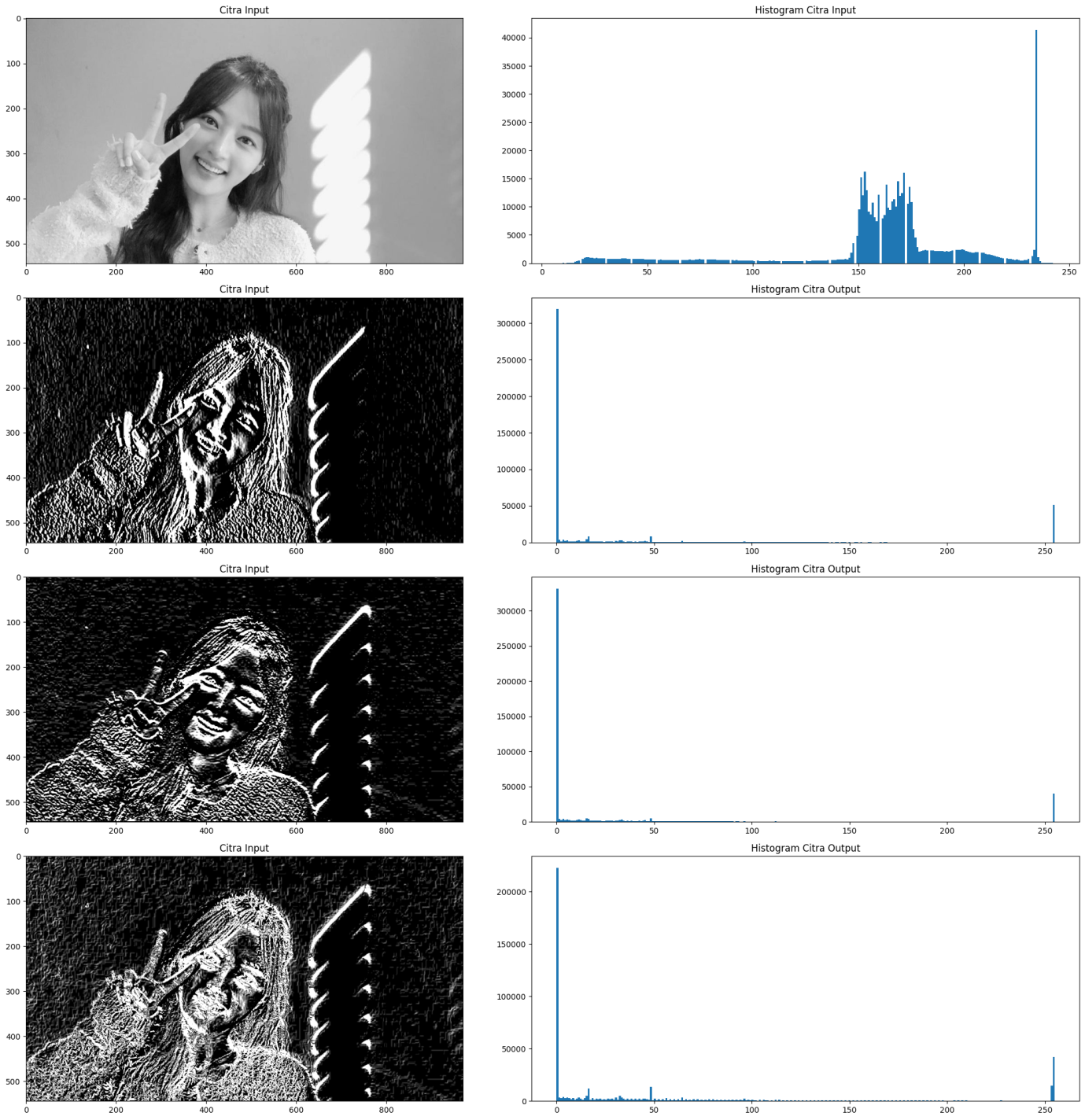
ax[6].set\_title ("Citra Input")

ax[7].hist (image\_sobel.ravel(), bins = 256)

ax[7].set\_title ("Histogram Citra Output")

fig.tight\_layout()

**Hasil :**

****

**Latihan 2 Filter Prewitt**

**Source Code :**

import cv2

import numpy as np

from matplotlib import pyplot as plt

from skimage import data

image = cv2.imread("songhayoon.jpg",0)

kernelx = np.array([[1,1,1], [0,0,0], [-1,-1,-1]])

kernely = np.array([[-1,0,1], [-1,0,1], [-1,0,1]])

image\_prewittx = cv2.filter2D(image, -1, kernelx)

image\_prewitty = cv2.filter2D(image, -1, kernely)

image\_prewitt = image\_prewittx + image\_prewitty

fig, axes = plt.subplots (4, 2, figsize = (20, 20))

ax = axes.ravel ()

ax[0].imshow(image, cmap = 'gray')

ax[0].set\_title ("Citra Input")

ax[1].hist (image.ravel(), bins = 256)

ax[1].set\_title ("Histogram Citra Input")

ax[2].imshow(image\_prewittx, cmap = 'gray')

ax[2].set\_title ("Citra Output Prewitt X")

ax[3].hist (image\_prewittx.ravel(), bins = 256)

ax[3].set\_title ("Histogram Citra Output Prewitt X")

ax[4].imshow(image\_prewitty, cmap = 'gray')

ax[4].set\_title ("Citra Output Prewitt Y")

ax[5].hist (image\_prewitty.ravel(), bins = 256)

ax[5].set\_title ("Histogram Citra Output Prewitt Y")

ax[6].imshow(image\_prewitt, cmap = 'gray')

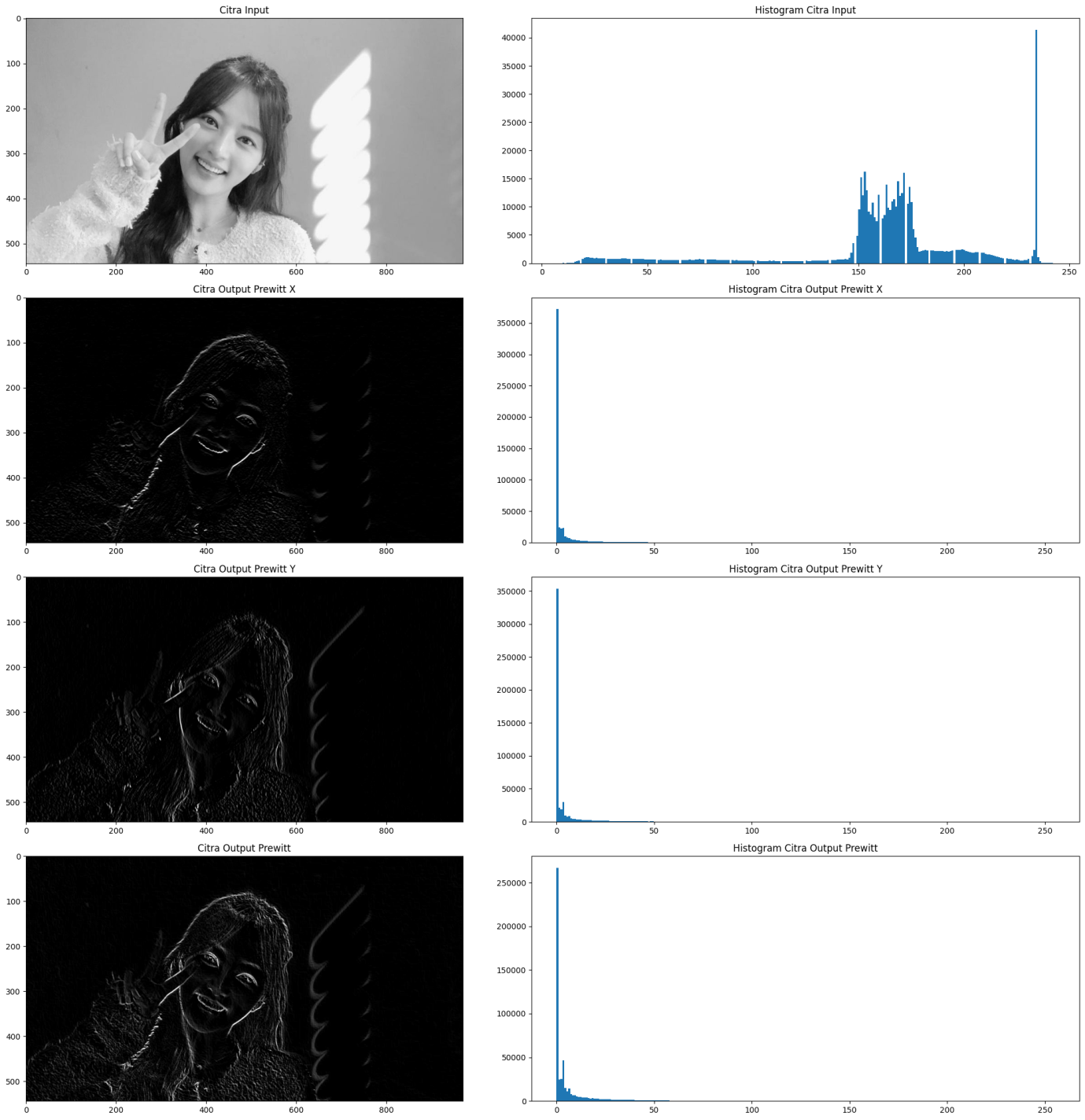
ax[6].set\_title ("Citra Output Prewitt")

ax[7].hist (image\_prewitt.ravel(), bins = 256)

ax[7].set\_title ("Histogram Citra Output Prewitt")

fig.tight\_layout()

**Hasil :**

****

**Latihan 3 Filter Canny**

**Source Code :**

import cv2

from matplotlib import pyplot as plt

image = cv2.imread('songhayoon.jpg',0)

img\_canny = cv2.Canny(image,100,200)

fig, axes = plt.subplots(2, 2, figsize=(6, 6))

ax = axes.ravel()

ax[0].imshow(image, cmap ='gray')

ax[0].set\_title("Citra Input")

ax[1].hist(image.ravel(), bins = 256)

ax[1].set\_title("Histogram Citra Input")

ax[2].imshow(img\_canny, cmap ='gray')

ax[2].set\_title("Citra Output")

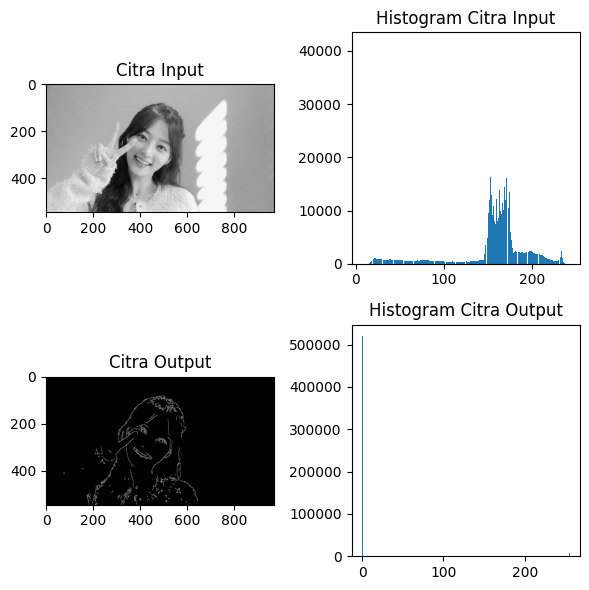
ax[3].hist(img\_canny.ravel(), bins = 256)

ax[3].set\_title("Histogram Citra Output")

fig.tight\_layout()

plt.show()

**Hasil :**

****

**Latihan 4 Laplacian**

**Source Code :**

import mahotas as mh

import numpy as np

from pylab import imshow, show

regions = np.zeros((8,8), bool)

regions[:3,:3] = 1

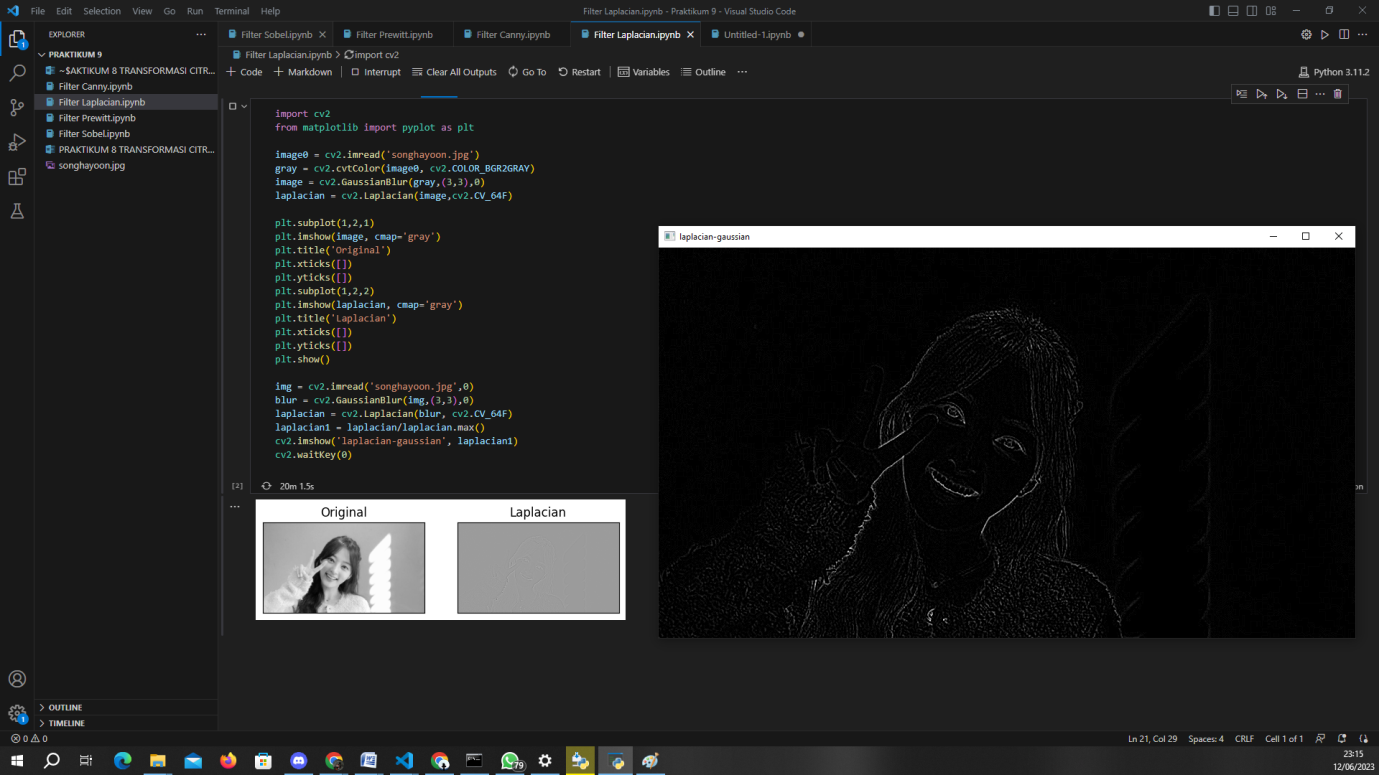
regions[6:,6:] = 1

labeled, nr\_objects = mh.label(regions)

imshow(labeled, interpolation='nearest')

show()

**Hasil :**

****

**Analisis :**

analisis singkat yang dapat saya uraikan mengenai beberapa filter populer dalam pengolahan citra: Sobel, Filter Prewitt, Filter Canny, dan Filter Laplacian.

1. Sobel Filter:
   * Sobel filter digunakan untuk mendeteksi tepi dalam citra.
   * Filter ini bekerja dengan menghitung gradien (perubahan intensitas) pada sumbu x dan y pada setiap piksel dalam citra.
   * Dalam penggunaannya, filter Sobel diterapkan pada citra grayscale atau pada setiap saluran warna dalam citra berwarna.
   * Hasil dari filter Sobel adalah dua citra terpisah yang mewakili gradien pada sumbu x dan y. Citra ini dapat digunakan untuk mengekstraksi fitur tepi atau untuk menghitung magnitude dan arah gradien.
   * Filter Sobel dapat digunakan secara individu atau digabungkan untuk menghasilkan gradien magnitude yang lebih akurat.
2. Filter Prewitt:
   * Filter Prewitt juga digunakan untuk deteksi tepi dalam citra, serupa dengan filter Sobel.
   * Filter ini bekerja dengan menghitung gradien pada sumbu x dan y dengan bobot tertentu.
   * Seperti Sobel, filter Prewitt juga dapat diterapkan pada citra grayscale atau saluran warna dalam citra berwarna.
   * Pada umumnya, filter Prewitt memberikan respons yang serupa dengan filter Sobel, tetapi dengan bobot yang berbeda.
3. Filter Canny:
   * Filter Canny adalah metode deteksi tepi yang lebih kompleks dan sering digunakan dalam pengolahan citra.
   * Filter ini memanfaatkan beberapa langkah, termasuk penghalusan citra dengan filter Gaussian, deteksi gradien menggunakan operator Sobel, penghilangan tepi yang tidak signifikan, dan penghubungan tepi yang kuat untuk membentuk garis tepi yang akurat.
   * Hasil dari filter Canny adalah citra dengan tepi yang tajam dan minim kesalahan deteksi.
4. Filter Laplacian:
   * Filter Laplacian digunakan untuk mengidentifikasi daerah dengan perubahan cepat dalam citra.
   * Filter ini bekerja dengan menghitung laplacian kedua (perubahan tingkat gradien) pada setiap piksel dalam citra.
   * Filter Laplacian dapat digunakan untuk mendeteksi tepi dan fitur tertentu dalam citra.
   * Namun, filter Laplacian cenderung sensitif terhadap kebisingan, sehingga biasanya diikuti dengan proses penghalusan untuk mengurangi efek noise.