Nama : Muhamad Arif Maulana

Nim : 1207070069

Tugas PCD

**Filter Rerata**

import matplotlib.pyplot as plt #mengimport modul matplotlib.pyplot untuk plotting

%matplotlib inline

from skimage import data #mengimport modul data dari paket skimage untuk mengakses dataset citra

from skimage.io import imread #mengimport fungsi imread dari modul io dalam paket skimage untuk membaca citra

from skimage.color import rgb2gray #mengimport fungsi rgb2gray dari modul color dalam paket skimage untuk mengkonversi citra RGB menjadi citra grayscale

import numpy as np #mengimport modul numpy dan menggunakan alias np untuk operasi numerik dan array

citra1 = imread(fname="foto/mobil.tif") #memanggil gambar dan menyimpannya ke variabel citra1

citra2 = imread(fname="foto/boneka2.tif") #memanggil gambar dan menyimpannya ke variabel citra2

print('Shape citra 1 : ', citra1.shape) #menampilkan bentuk (shape) citra1 yang merupakan dimensi citra

print('Shape citra 2 : ', citra2.shape) #menampilkan bentuk (shape) citra2 yang merupakan dimensi citra

fig, axes = plt.subplots(1, 2, figsize=(10, 10)) #membuat sebuah objek gambar (figure) dengan 1 baris dan 2 kolom, dan mengatur ukuran figur menjadi 10x10

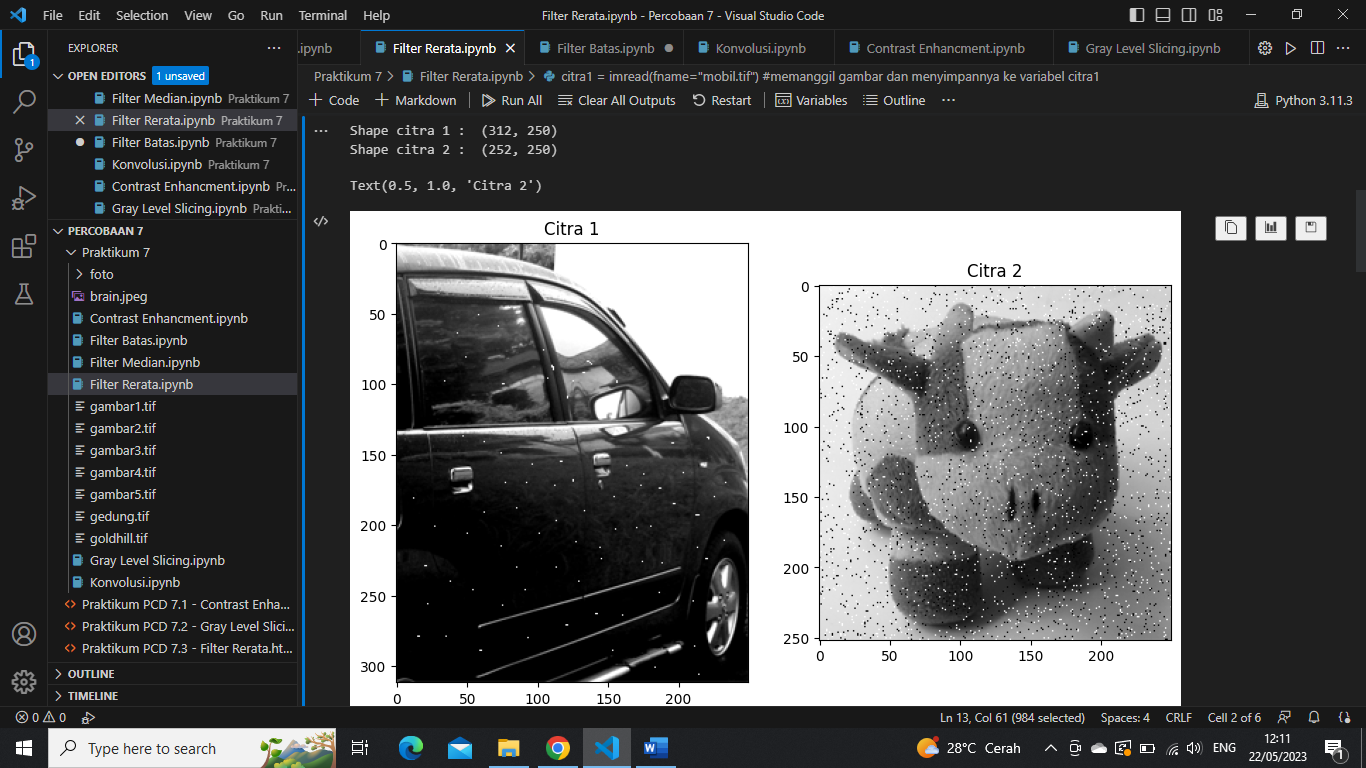
ax = axes.ravel() #mengubah array 2D dari axes menjadi array 1D untuk mengakses setiap sumbu plot secara individu

ax[0].imshow(citra1, cmap = 'gray') #menampilkan citra1 pada sumbu pertama dengan colormap gray

ax[0].set\_title("Citra 1") #mengatur judul untuk sumbu pertama

ax[1].imshow(citra2, cmap = 'gray') #menampilkan citra2 pada sumbu kedua dengan colormap gray

ax[1].set\_title("Citra 2") #mengatur judul untuk sumbu kedua



copyCitra1 = citra1.copy().astype(float) #membuat salinan citra citra1 dan mengubah tipe datanya menjadi float

copyCitra2 = citra2.copy().astype(float) #membuat salinan citra citra2 dan mengubah tipe datanya menjadi float.

m1,n1 = copyCitra1.shape #mendapatkan dimensi citra1 dan menyimpannya dalam variabel m1 dan n1

output1 = np.empty([m1, n1]) #membuat array kosong dengan dimensi yang sama dengan citra1 untuk menyimpan hasil pemrosesan

m2,n2 = copyCitra2.shape #mendapatkan dimensi citra2 dan menyimpannya dalam variabel m2 dan n2

output2 = np.empty([m2, n2]) #membuat array kosong dengan dimensi yang sama dengan citra2 untuk menyimpan hasil pemrosesan

print('Shape copy citra 1 : ', copyCitra1.shape) #menampilkan bentuk (shape) dari salinan citra1

print('Shape output citra 1 : ', output1.shape) #menampilkan bentuk (shape) dari output1

print('m1 : ',m1) #menampilkan nilai m1 yang merupakan jumlah baris citra1

print('n1 : ',n1) #menampilkan nilai n1 yang merupakan jumlah kolom citra1

print() #mencetak baris kosong

print('Shape copy citra 2 : ', copyCitra2.shape) #menampilkan bentuk (shape) dari salinan citra2

print('Shape output citra 3 : ', output2.shape) #menampilkan bentuk (shape) dari output2

print('m2 : ',m2) #menampilkan nilai m2 yang merupakan jumlah baris citra2

print('n2 : ',n2) #menampilkan nilai n2 yang merupakan jumlah kolom citra2

print() #mencetak baris kosong

for baris in range(0, m1-1): #memulai loop untuk setiap baris dalam citra1

    for kolom in range(0, n1-1): #memulai loop untuk setiap kolom dalam citra1

        a1 = baris #menyimpan indeks baris saat ini dalam variabel a1

        b1 = kolom #menyimpan indeks kolom saat ini dalam variabel b1

        jumlah = copyCitra1[a1-1, b1-1] + copyCitra1[a1-1, b1] + copyCitra1[a1-1, b1-1] + \

                 copyCitra1[a1, b1-1] + copyCitra1[a1, b1] + copyCitra1[a1, b1+1] + \

                 copyCitra1[a1+1, b1-1] + copyCitra1[a1+1, b1] + copyCitra1[a1+1, b1+1]; #menghitung jumlah piksel sekitar dalam citra1 menggunakan kernel 3x3

        output1[a1, b1] = (1/9 \* jumlah) #menyimpan nilai rata-rata piksel sekitar dalam output1

for baris1 in range(0, m2-1): #memulai loop untuk setiap baris dalam citra2

    for kolom1 in range(0, n2-1): #memulai loop untuk setiap kolom dalam citra2

        a1 = baris1 #menyimpan indeks baris saat ini dalam variabel a1

        b1 = kolom1 #menyimpan indeks kolom saat ini dalam variabel b1

        jumlah = copyCitra2[a1-1, b1-1] + copyCitra2[a1-1, b1] + copyCitra2[a1-1, b1-1] + \

                 copyCitra2[a1, b1-1] + copyCitra2[a1, b1] + copyCitra2[a1, b1+1] + \

                 copyCitra2[a1+1, b1-1] + copyCitra2[a1+1, b1] + copyCitra2[a1+1, b1+1]; ##menghitung jumlah piksel sekitar dalam citra2 menggunakan kernel 3x3

        output2[a1, b1] = (1/9 \* jumlah) #menyimpan nilai rata-rata piksel sekitar dalam output2

fig, axes = plt.subplots(2, 2, figsize=(9, 9)) #membuat sebuah objek gambar (figure) dengan 2 baris dan 2 kolom, dan mengatur ukuran figur menjadi 9x9

ax = axes.ravel()#mengubah array 2D dari axes menjadi array 1D untuk mengakses setiap sumbu plot secara individu

ax[0].imshow(citra1, cmap = 'gray') #menampilkan citra1 pada sumbu pertama dengan colormap gray

ax[0].set\_title("Input Citra 1") #mengatur judul untuk sumbu pertama

ax[1].imshow(citra2, cmap = 'gray') #menampilkan citra2 pada sumbu kedua dengan colormap gray

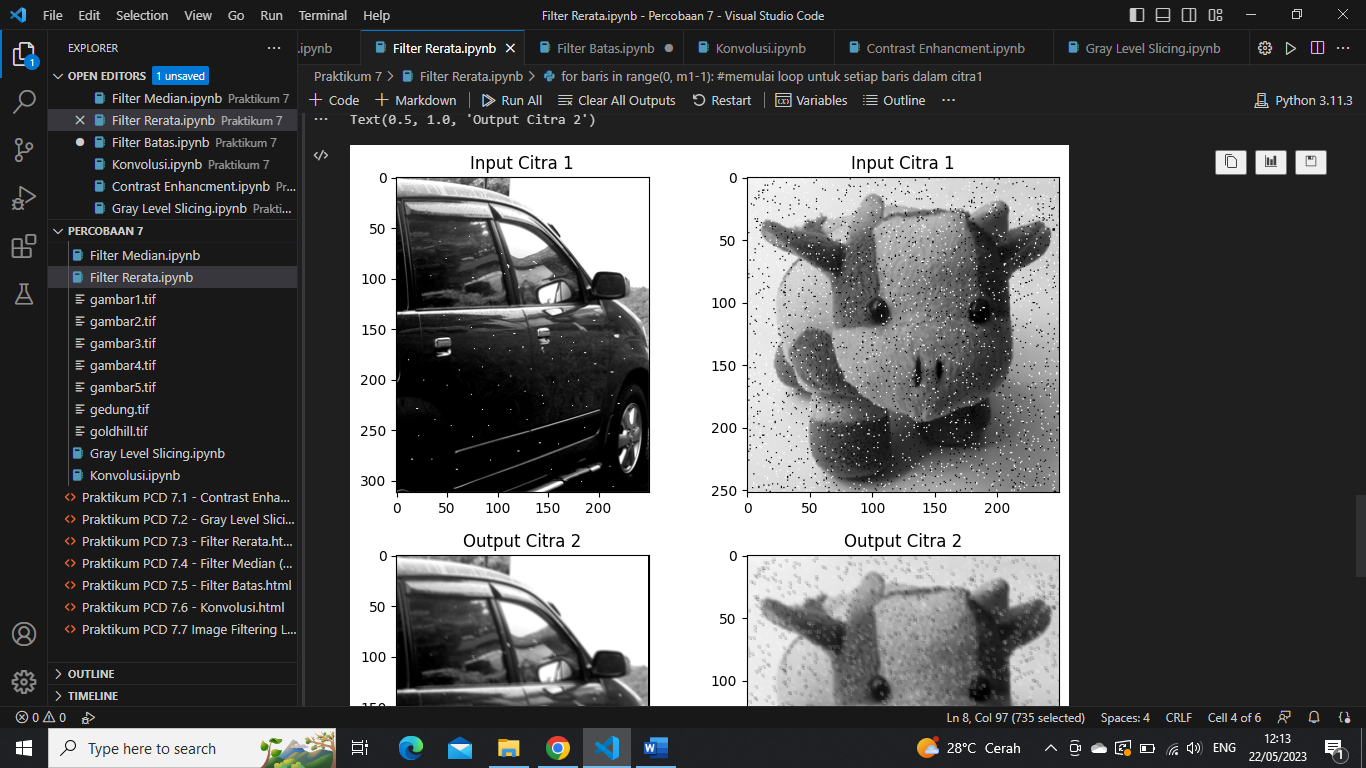
ax[1].set\_title("Input Citra 1") #mengatur judul untuk sumbu kedua

ax[2].imshow(output1, cmap = 'gray') #menampilkan output1 pada sumbu ketiga dengan colormap gray

ax[2].set\_title("Output Citra 2") #mengatur judul untuk sumbu ketiga

ax[3].imshow(output2, cmap = 'gray') #menampilkan output2 pada sumbu keempat dengan colormap gray

ax[3].set\_title("Output Citra 2") #mengatur judul untuk sumbu keempat



**Filter Median**

import matplotlib.pyplot as plt #mengimport modul matplotlib.pyplot untuk plotting

%matplotlib inline

from skimage import data #mengimport modul data dari paket skimage untuk mengakses dataset citra

from skimage.io import imread #mengimport fungsi imread dari modul io dalam paket skimage untuk membaca citra

from skimage.color import rgb2gray #mengimport fungsi rgb2gray dari modul color dalam paket skimage untuk mengkonversi citra RGB menjadi citra grayscale

import numpy as np #mengimport modul numpy dan menggunakan alias np untuk operasi numerik dan array

import cv2 #mengimport modul cv2

citra1 = imread(fname="foto/gambar3.jpg") #memanggil gambar dan menyimpannya ke variabel citra1

citra1 = cv2.cvtColor(citra1,cv2.COLOR\_RGB2GRAY)

citra2 = imread(fname="foto/gambar5.jpg") #memanggil gambar dan menyimpannya ke variabel citra2

citra2 = cv2.cvtColor(citra2,cv2.COLOR\_RGB2GRAY)

print('Shape citra 1 : ', citra1.shape) #menampilkan bentuk (shape) citra1 yang merupakan dimensi citra

print('Shape citra 2 : ', citra2.shape) #menampilkan bentuk (shape) citra2 yang merupakan dimensi citra

fig, axes = plt.subplots(1, 2, figsize=(10, 10)) #membuat sebuah objek gambar (figure) dengan 1 baris dan 2 kolom, dan mengatur ukuran figur menjadi 10x10

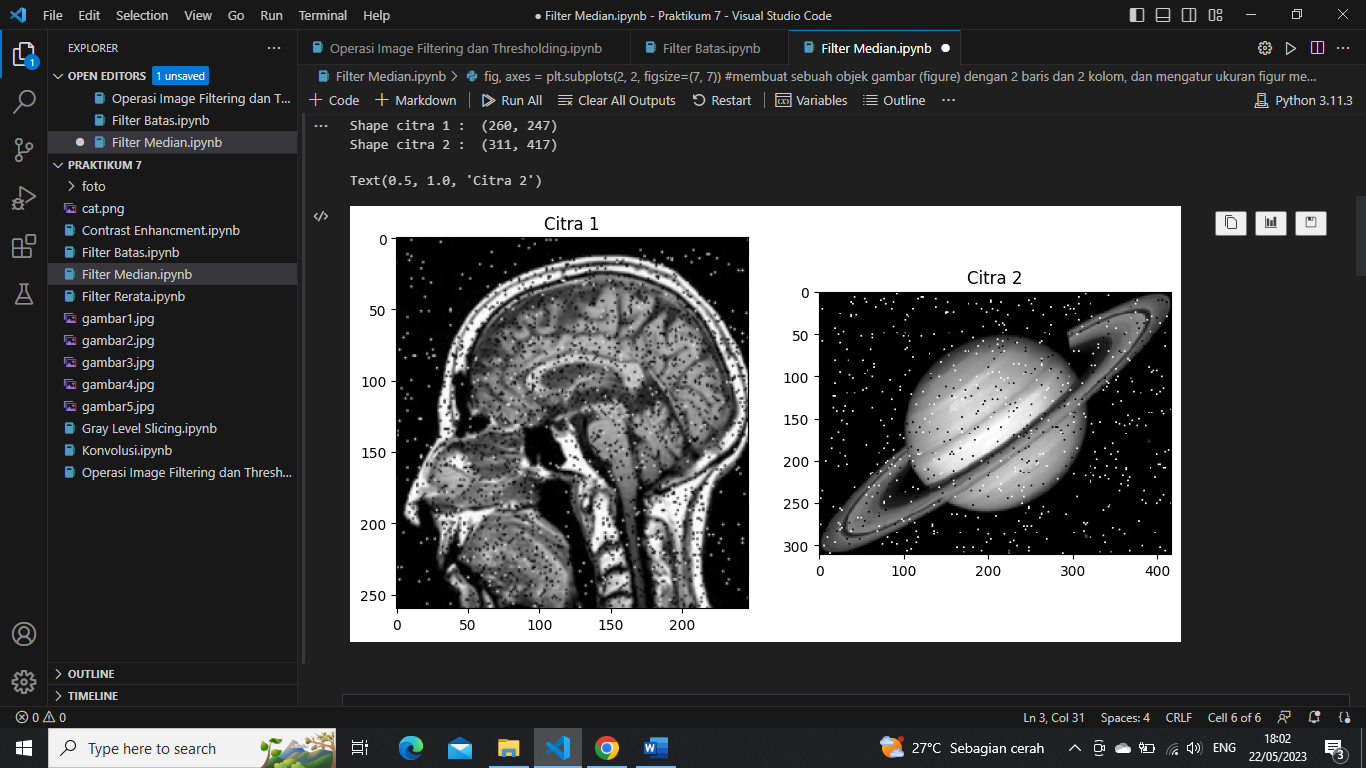
ax = axes.ravel() #mengubah array 2D dari axes menjadi array 1D untuk mengakses setiap sumbu plot secara individu

ax[0].imshow(citra1, cmap = 'gray') #menampilkan citra1 pada sumbu pertama dengan colormap gray

ax[0].set\_title("Citra 1") #mengatur judul untuk sumbu pertama

ax[1].imshow(citra2, cmap = 'gray') #menampilkan citra2 pada sumbu kedua dengan colormap gray

ax[1].set\_title("Citra 2") #mengatur judul untuk sumbu kedua



copyCitra1 = citra1.copy() #membuat salinan citra citra1

copyCitra2 = citra2.copy() #membuat salinan citra citra2

m1,n1 = copyCitra1.shape #mendapatkan dimensi citra1 dan menyimpannya dalam variabel m1 dan n1

output1 = np.empty([m1, n1]) #membuat array kosong dengan dimensi yang sama dengan citra1 untuk menyimpan hasil pemrosesan

m2,n2 = copyCitra2.shape #mendapatkan dimensi citra2 dan menyimpannya dalam variabel m2 dan n2

output2 = np.empty([m2, n2]) #membuat array kosong dengan dimensi yang sama dengan citra2 untuk menyimpan hasil pemrosesan

print('Shape copy citra 1 : ', copyCitra1.shape) #menampilkan bentuk (shape) dari salinan citra1

print('Shape output citra 1 : ', output1.shape) #menampilkan bentuk (shape) dari output1

print('m1 : ',m1) #menampilkan nilai m1 yang merupakan jumlah baris citra1

print('n1 : ',n1) #menampilkan nilai n1 yang merupakan jumlah kolom citra1

print() #mencetak baris kosong

print('Shape copy citra 2 : ', copyCitra2.shape) #menampilkan bentuk (shape) dari salinan citra2

print('Shape output citra 2 : ', output2.shape) #menampilkan bentuk (shape) dari output2

print('m2 : ',m2) #menampilkan nilai m2 yang merupakan jumlah baris citra2

print('n2 : ',n2) #menampilkan nilai n2 yang merupakan jumlah kolom citra2

print() #mencetak baris kosong

for baris in range(0, m1-1): #memulai loop untuk setiap baris dalam citra1

    for kolom in range(0, n1-1): #memulai loop untuk setiap kolom dalam citra1

        a1 = baris #menyimpan indeks baris saat ini dalam variabel a1

        b1 = kolom #menyimpan indeks kolom saat ini dalam variabel b1

        dataA = [copyCitra1[a1-1, b1-1], copyCitra1[a1-1, b1], copyCitra1[a1-1, b1+1], \

              copyCitra1[a1, b1-1], copyCitra1[a1, b1], copyCitra1[a1, b1+1], \

              copyCitra1[a1+1, b1-1], copyCitra1[a1+1, b1], copyCitra1[a1+1, b1+1]] #membuat sebuah list dataA pada citra2

        for i in range(1, 8): #melakukan loop dari 1 hingga 7. Variabel i

            for j in range(i, 9): #melakukan loop dari nilai i hingga 8. Variabel j

                if dataA[i] > dataA[j]: #memeriksa apakah nilai dataA[i] lebih besar dari dataA[j]

                    tmpA = dataA[i]; #menyimpan nilai dataA[i] dalam variabel tmpA

                    dataA[i] = dataA[j]; #mengganti nilai dataA[i] dengan nilai dataA[j]

                    dataA[j]= tmpA; #mengganti nilai dataA[j] dengan nilai tmpA yang sebelumnya disimpan

        output1[a1, b1] = dataA[5] #menyimpan nilai dataA[5] dalam output1[a1, b1]

for baris1 in range(0, m2-1): #memulai loop untuk setiap baris dalam citra2

    for kolom1 in range(0, n2-1): #memulai loop untuk setiap kolom dalam citra2

        a1 = baris1 #menyimpan indeks baris saat ini dalam variabel a1

        b1 = kolom1 #menyimpan indeks kolom saat ini dalam variabel b1

        dataA = [copyCitra2[a1-1, b1-1], copyCitra2[a1-1, b1], copyCitra2[a1-1, b1+1], \

              copyCitra2[a1, b1-1], copyCitra2[a1, b1], copyCitra2[a1, b1+1], \

              copyCitra2[a1+1, b1-1], copyCitra2[a1+1, b1], copyCitra2[a1+1, b1+1]] #membuat sebuah list dataA pada citra2

        for i in range(1, 8): #melakukan loop dari 1 hingga 7. Variabel i

            for j in range(i, 9): #melakukan loop dari nilai i hingga 8. Variabel j

                if dataA[i] > dataA[j]: #memeriksa apakah nilai dataA[i] lebih besar dari dataA[j]

                    tmpA = dataA[i]; #menyimpan nilai dataA[i] dalam variabel tmpA

                    dataA[i] = dataA[j]; #mengganti nilai dataA[i] dengan nilai dataA[j]

                    dataA[j]= tmpA; #mengganti nilai dataA[j] dengan nilai tmpA yang sebelumnya disimpan

        output2[a1, b1] = dataA[5] #menyimpan nilai dataA[5] dalam output1[a1, b1]

fig, axes = plt.subplots(2, 2, figsize=(7, 7)) #membuat sebuah objek gambar (figure) dengan 2 baris dan 2 kolom, dan mengatur ukuran figur menjadi 9x9

ax = axes.ravel()#mengubah array 2D dari axes menjadi array 1D untuk mengakses setiap sumbu plot secara individu

ax[0].imshow(citra1, cmap = 'gray') #menampilkan citra1 pada sumbu pertama dengan colormap gray

ax[0].set\_title("Input Citra 1") #mengatur judul untuk sumbu pertama

ax[1].imshow(citra2, cmap = 'gray') #menampilkan citra2 pada sumbu kedua dengan colormap gray

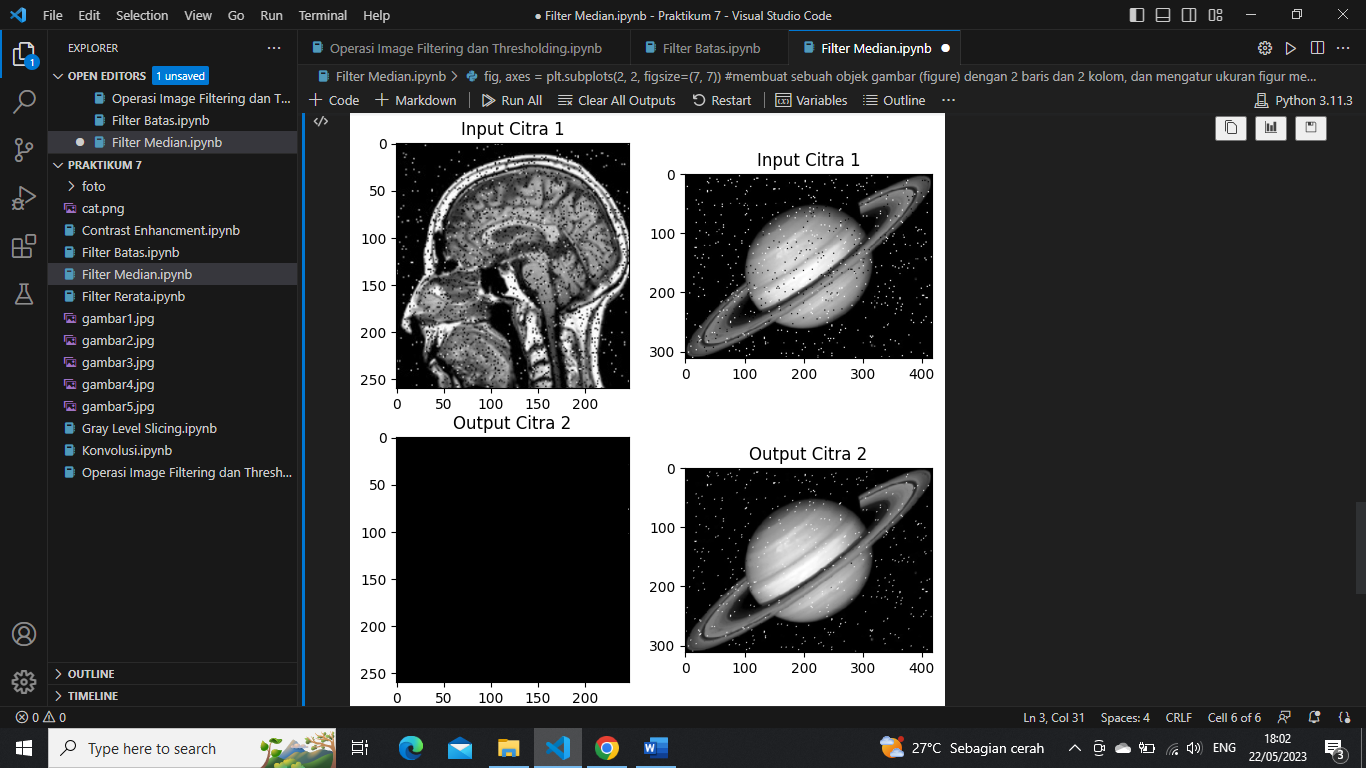
ax[1].set\_title("Input Citra 1") #mengatur judul untuk sumbu kedua

ax[2].imshow(output1, cmap = 'gray') #menampilkan output1 pada sumbu ketiga dengan colormap gray

ax[2].set\_title("Output Citra 2") #mengatur judul untuk sumbu ketiga

ax[3].imshow(output2, cmap = 'gray') #menampilkan output2 pada sumbu keempat dengan colormap gray

ax[3].set\_title("Output Citra 2") #mengatur judul untuk sumbu keempat



**Filter Batas**

import matplotlib.pyplot as plt #mengimport modul matplotlib.pyplot untuk plotting

%matplotlib inline

from skimage import data #mengimport modul data dari paket skimage untuk mengakses dataset citra

from skimage.io import imread #mengimport fungsi imread dari modul io dalam paket skimage untuk membaca citra

from skimage.color import rgb2gray #mengimport fungsi rgb2gray dari modul color dalam paket skimage untuk mengkonversi citra RGB menjadi citra grayscale

import numpy as np #mengimport modul numpy dan menggunakan alias np untuk operasi numerik dan array

import cv2 #mengimport modul cv2

citra1 = imread(fname="foto/gambar1.jpg") #memanggil gambar dan menyimpannya ke variabel citra1

citra1 = cv2.cvtColor(citra1,cv2.COLOR\_RGB2GRAY)

citra2 = imread(fname="foto/gambar2.jpg") #memanggil gambar dan menyimpannya ke variabel citra2

citra2 = cv2.cvtColor(citra2,cv2.COLOR\_RGB2GRAY)

print('Shape citra 1 : ', citra1.shape) #menampilkan bentuk (shape) citra1 yang merupakan dimensi citra

print('Shape citra 2 : ', citra2.shape) #menampilkan bentuk (shape) citra2 yang merupakan dimensi citra

fig, axes = plt.subplots(1, 2, figsize=(10, 10)) #membuat sebuah objek gambar (figure) dengan 1 baris dan 2 kolom, dan mengatur ukuran figur menjadi 10x10

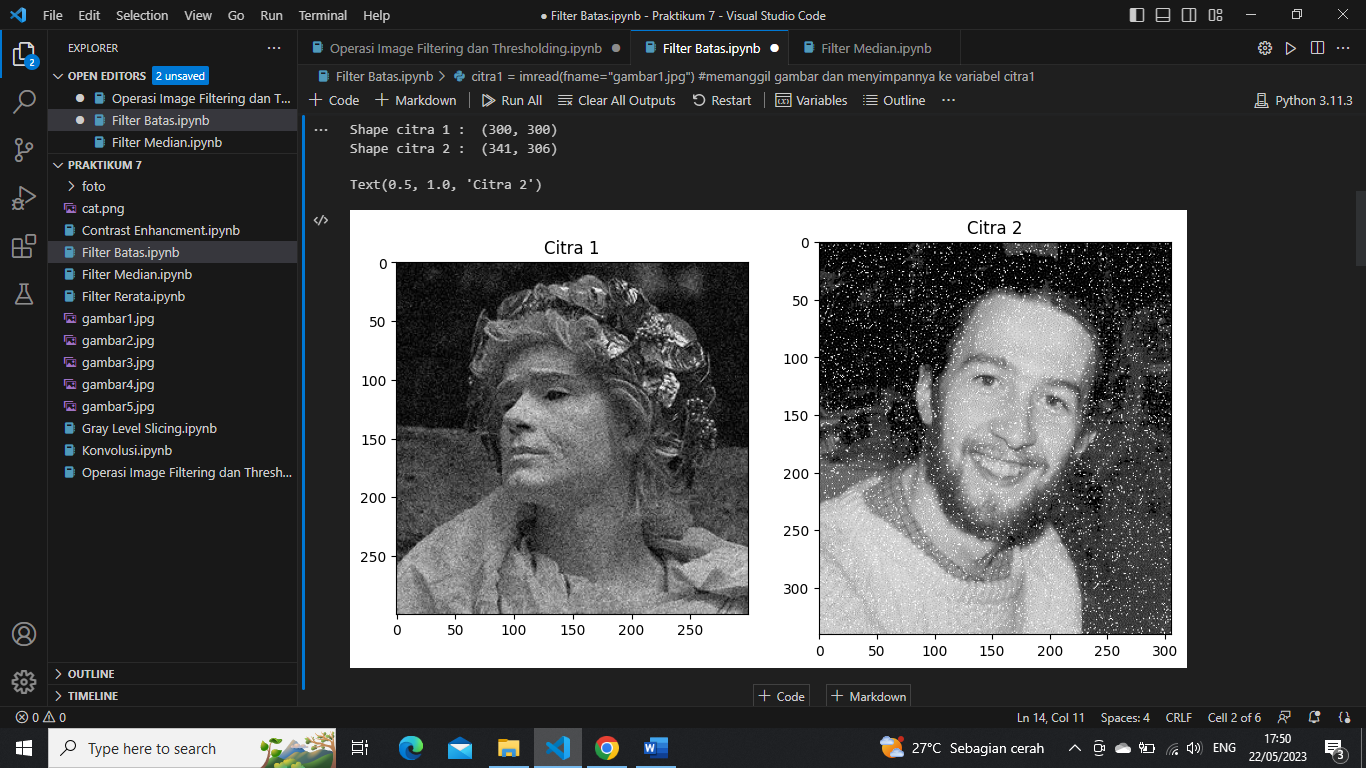
ax = axes.ravel() #mengubah array 2D dari axes menjadi array 1D untuk mengakses setiap sumbu plot secara individu

ax[0].imshow(citra1, cmap = 'gray') #menampilkan citra1 pada sumbu pertama dengan colormap gray

ax[0].set\_title("Citra 1") #mengatur judul untuk sumbu pertama

ax[1].imshow(citra2, cmap = 'gray') #menampilkan citra2 pada sumbu kedua dengan colormap gray

ax[1].set\_title("Citra 2") #mengatur judul untuk sumbu kedua



copyCitra1 = citra1.copy() #membuat salinan citra

copyCitra2 = citra2.copy() #membuat salinan citra

m1,n1 = copyCitra1.shape #mendapatkan dimensi citra1 dan menyimpannya dalam variabel m1 dan n1

output1 = np.empty([m1, n1]) #membuat array kosong dengan dimensi yang sama dengan citra1 untuk menyimpan hasil pemrosesan

m2,n2 = copyCitra2.shape #mendapatkan dimensi citra2 dan menyimpannya dalam variabel m2 dan n2

output2 = np.empty([m2, n2]) #membuat array kosong dengan dimensi yang sama dengan citra2 untuk menyimpan hasil pemrosesan

print('Shape copy citra 1 : ', copyCitra1.shape) #menampilkan bentuk (shape) dari salinan citra1

print('Shape output citra 1 : ', output1.shape) #menampilkan bentuk (shape) dari output1

print('m1 : ',m1) #menampilkan nilai m1 yang merupakan jumlah baris citra1

print('n1 : ',n1) #menampilkan nilai n1 yang merupakan jumlah kolom citra1

print() #mencetak baris kosong

print('Shape copy citra 2 : ', copyCitra2.shape) #menampilkan bentuk (shape) dari salinan citra2

print('Shape output citra 3 : ', output2.shape) #menampilkan bentuk (shape) dari output2

print('m2 : ',m2) #menampilkan nilai m2 yang merupakan jumlah baris citra2

print('n2 : ',n2) #menampilkan nilai n2 yang merupakan jumlah kolom citra2

print() #mencetak baris kosong

for baris in range(0, m1-1): #memulai loop untuk setiap baris dalam citra1

    for kolom in range(0, n1-1): #memulai loop untuk setiap kolom dalam citra1

        a1 = baris #menyimpan indeks baris saat ini dalam variabel a1

        b1 = kolom #menyimpan indeks kolom saat ini dalam variabel b1

        arr = np.array([copyCitra1[a1-1, b1-1], copyCitra1[a1-1, b1], copyCitra1[a1, b1+1], \

            copyCitra1[a1, b1-1], copyCitra1[a1, b1+1], copyCitra1[a1+1, b1-1],  \

            copyCitra1[a1+1, b1], copyCitra1[a1+1, b1+1]]) #membuat sebuah array numpy (arr) dari copyCitra1

        minPiksel = np.amin(arr); #menghitung nilai piksel minimum dari array arr

        maksPiksel = np.amax(arr); #menghitung nilai piksel maksimum dari array arr

        if copyCitra1[baris, kolom] < minPiksel : #jika nilai piksel pada posisi (baris, kolom) kemudian pada copyCitra1 lebih kecil daripada minPiksel maka :

            output1[baris, kolom] = minPiksel #mengatur nilai piksel pada posisi (baris, kolom) pada output1 menjadi minPiksel

        else :

            if copyCitra1[baris, kolom] > maksPiksel : #jika nilai piksel pada posisi (baris, kolom) kemudian pada copyCitra1 lebih besar daripada minPiksel maka :

                output1[baris, kolom] = maksPiksel #mengatur nilai piksel pada posisi (baris, kolom) pada output1 menjadi maksPiksel

            else :

                output1[baris, kolom] = copyCitra1[baris, kolom] #mengatur nilai piksel pada posisi (baris, kolom) pada output1 menjadi nilai piksel pada posisi yang sama pada copyCitra1

for baris1 in range(0, m2-1): #memulai loop untuk setiap baris dalam citra1

    for kolom1 in range(0, n2-1): #memulai loop untuk setiap kolom dalam citra1

        a1 = baris1 #menyimpan indeks kolom saat ini dalam variabel b1

        b1 = kolom1 #menyimpan indeks kolom saat ini dalam variabel b1

        arr = np.array([copyCitra2[a1-1, b1-1], copyCitra2[a1-1, b1], copyCitra2[a1, b1+1], \

            copyCitra2[a1, b1-1], copyCitra2[a1, b1+1], copyCitra2[a1+1, b1-1],  \

            copyCitra2[a1+1, b1], copyCitra2[a1+1, b1+1]]) #membuat sebuah array numpy (arr) dari copyCitra2

        minPiksel = np.amin(arr); #menghitung nilai piksel minimum dari array arr

        maksPiksel = np.amax(arr); #menghitung nilai piksel maksimal dari array arr

        if copyCitra2[baris1, kolom1] < minPiksel : #jika nilai piksel pada posisi (baris, kolom) kemudian pada copyCitra2 lebih kecil daripada minPiksel maka :

            output2[baris1, kolom1] = minPiksel #mengatur nilai piksel pada posisi (baris, kolom) pada output2 menjadi minPiksel

        else :

            if copyCitra2[baris1, kolom1] > maksPiksel : #jika nilai piksel pada posisi (baris, kolom) kemudian pada copyCitra1 lebih besar daripada minPiksel maka :

                output2[baris1, kolom1] = maksPiksel #mengatur nilai piksel pada posisi (baris, kolom) pada output1 menjadi maksPiksel

            else :

                output2[baris1, kolom1] = copyCitra2[baris1, kolom1] #mengatur nilai piksel pada posisi (baris, kolom) pada output1 menjadi nilai piksel pada posisi yang sama pada copyCitra1

fig, axes = plt.subplots(2, 2, figsize=(8, 8)) #membuat figure dan array axes dengan ukuran 2x2 dan ukuran total figure

ax = axes.ravel() #membentuk array ax dengan menggunakan metode ravel() pada array axes

ax[0].imshow(citra1, cmap = 'gray') #menampilkan citra1 pada subplot pertama menggunakan metode imshow()

ax[0].set\_title("Input Citra 1") #menamai gambar

ax[1].imshow(citra2, cmap = 'gray') #menampilkan citra2 pada subplot pertama menggunakan metode imshow()

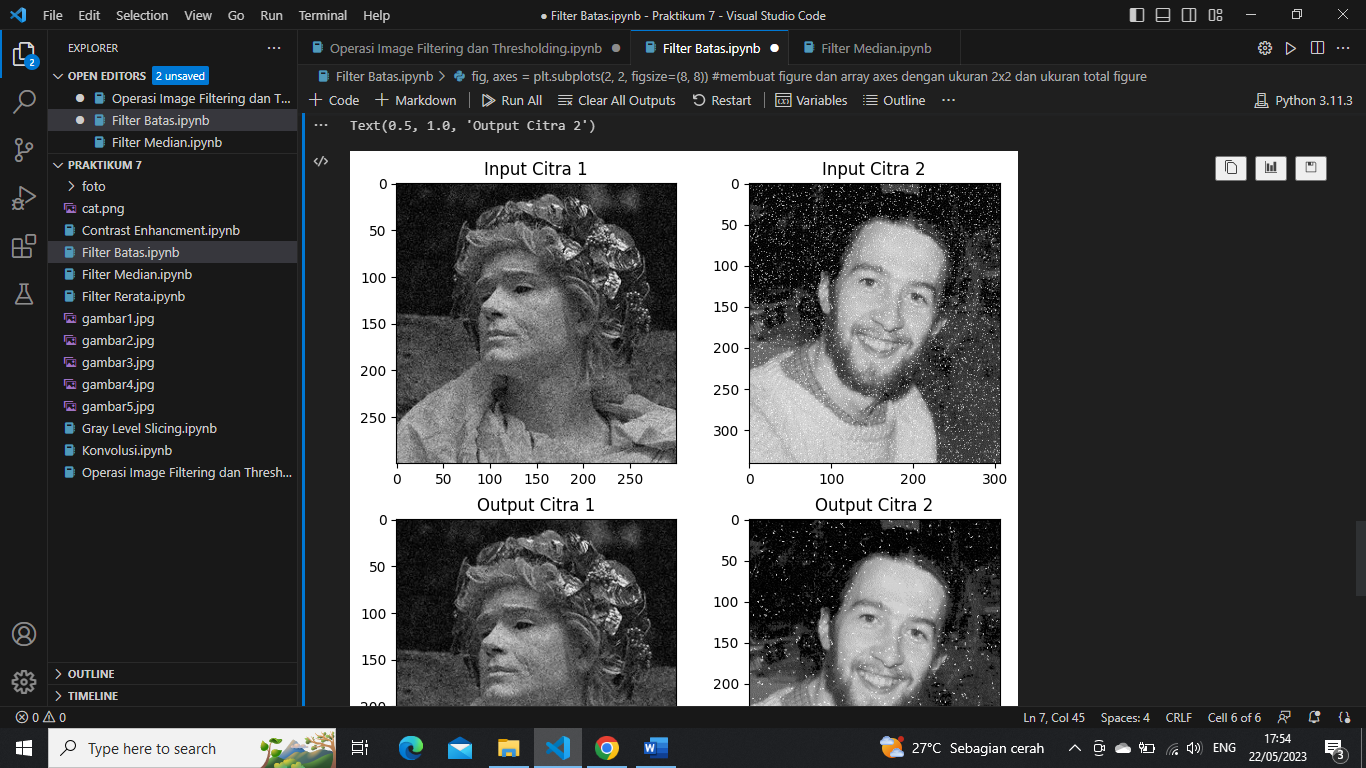
ax[1].set\_title("Input Citra 2") #menamai gambar

ax[2].imshow(output1, cmap = 'gray') #menampilkan output1 pada subplot pertama menggunakan metode imshow()

ax[2].set\_title("Output Citra 1") #menamai gambar

ax[3].imshow(output2, cmap = 'gray') #menampilkan output2 pada subplot pertama menggunakan metode imshow()

ax[3].set\_title("Output Citra 2") #menamai gambar



**Konvolusi**

import matplotlib.pyplot as plt #mengimport modul matplotlib.pyplot untuk plotting

%matplotlib inline

from skimage import data #mengimport modul data dari paket skimage untuk mengakses dataset citra

from skimage.io import imread #mengimport fungsi imread dari modul io dalam paket skimage untuk membaca citra

from skimage.color import rgb2gray #mengimport fungsi rgb2gray dari modul color dalam paket skimage untuk mengkonversi citra RGB menjadi citra grayscale

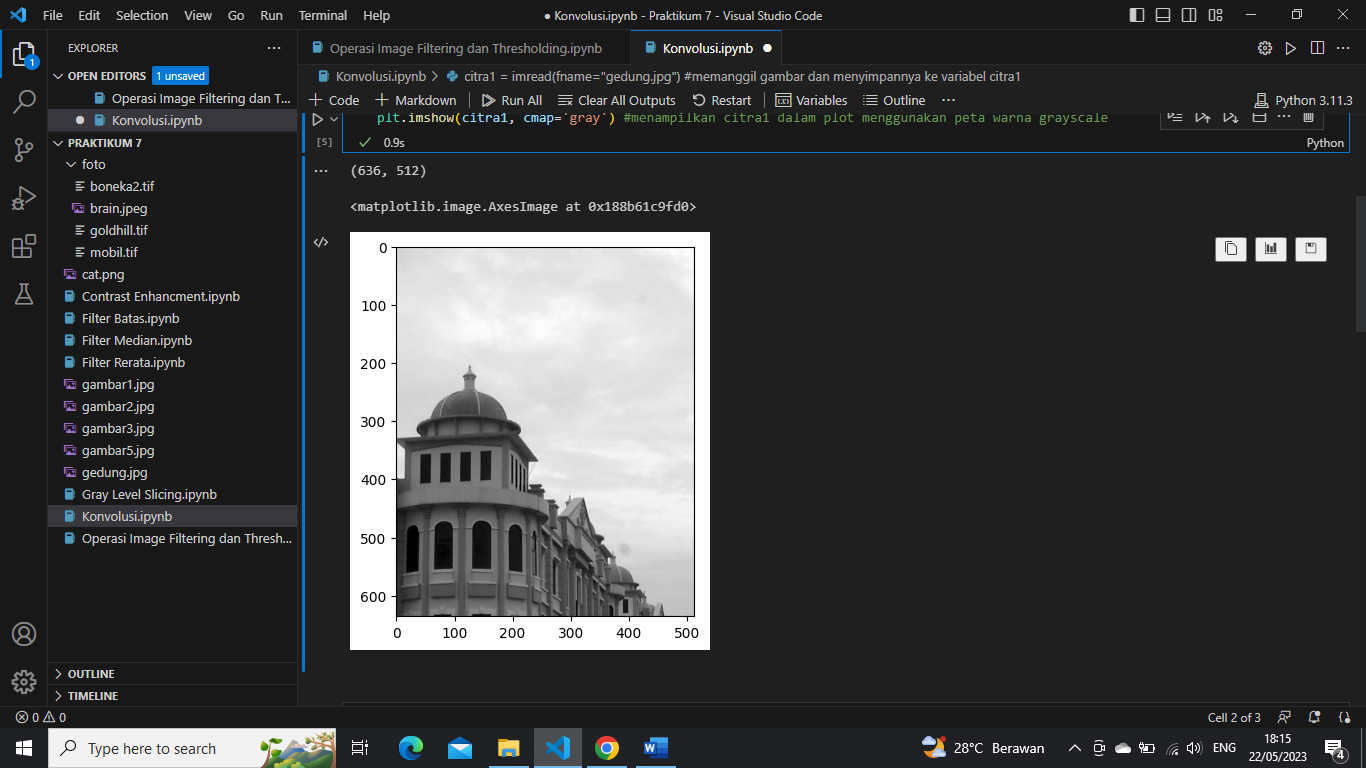
import numpy as np #mengimport modul numpy dan menggunakan alias np untuk operasi numerik dan array

import cv2 #mengimport modul cv2

citra1 = imread(fname="foto/gedung.jpg") #memanggil gambar dan menyimpannya ke variabel citra1

print(citra1.shape) #encetak bentuk (shape) dari citra1

plt.imshow(citra1, cmap='gray') #menampilkan citra1 dalam plot menggunakan peta warna grayscale



kernel = np.array([[-1, 0, -1],

                   [0, 4, 0],

                   [-1, 0, -1]]) #membentuk kernel atau matriks filter yang akan digunakan dalam proses filtering citra

citraOutput = cv2.filter2D(citra1, -1, kernel)#menerapkan operasi filtering pada citra

fig, axes = plt.subplots(1, 2, figsize=(12, 12)) #membuat plot dengan ukuran (figsize) 12x12 dan 1 baris serta 2 kolom subplot

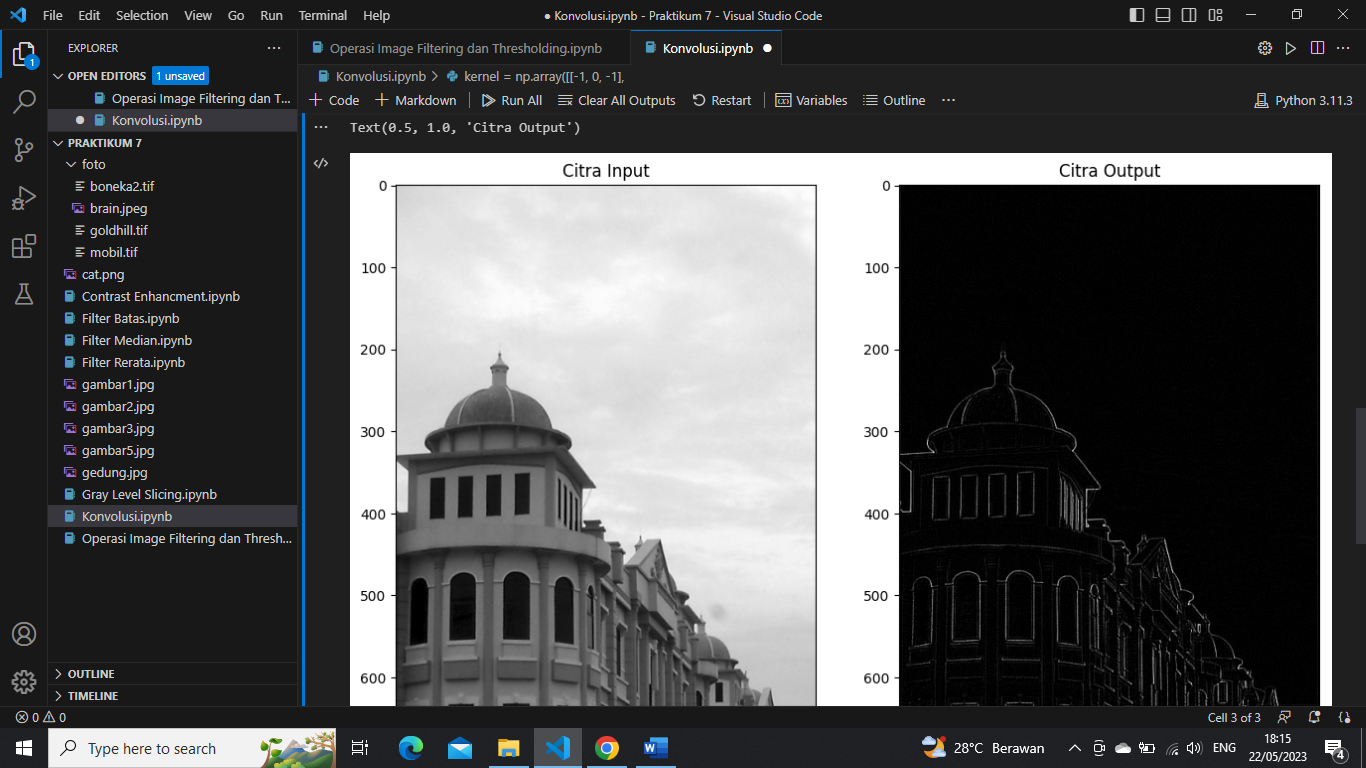
ax = axes.ravel()

ax[0].imshow(citra1, cmap = 'gray') #menampilkan citra input citra1 menggunakan peta warna grayscale

ax[0].set\_title("Citra Input") #memberi judul gambar

ax[1].imshow(citraOutput, cmap = 'gray') #menampilkan citra input citra1 menggunakan peta warna grayscale

ax[1].set\_title("Citra Output") #memberi judul gambar



**Contrast Enhancment**

import numpy as np #mengimport modul numpy dan menggunakan alias np untuk operasi numerik dan array

import matplotlib.pyplot as plt #mengimport modul matplotlib.pyplot untuk plotting

%matplotlib inline

import cv2 #mengimport modul cv2

import matplotlib.image as mpimg #mengimport modul matplotlib.image

from skimage import data #mengimport modul data dari paket skimage untuk mengakses dataset citra

image = data.camera() #mengambil citra kamera bawaan dari library skimage yang disebut "camera"

image\_equalized = cv2.equalizeHist(image) #melakukan ekualisasi histogram pada citra image

clahe = cv2.createCLAHE(clipLimit=2, tileGridSize=(8,8)) #membuat objek CLAHE (Contrast Limited Adaptive Histogram Equalization)

image\_clahe = clahe.apply(image) #menerapkan CLAHE pada citra image

image\_cs = np.zeros((image.shape[0],image.shape[1]),dtype = 'uint8') #membuat aray kosong

min = np.min(image) #menghitung nilai minimum

max = np.max(image) #menghitung nilai maksimum

for i in range(image.shape[0]): #melakukan iterasi untuk setiap piksel dalam citra image

    for j in range(image.shape[1]):

        image\_cs[i,j] = 255\*(image[i,j]-min)/(max-min) #menghitung nilai piksel baru dalam citra hasil kontras stretching

copyCamera = image.copy().astype(float) #membuat salinan citra image dengan tipe data float

m1,n1 = copyCamera.shape #mengambil ukuran citra copyCamera

output1 = np.empty([m1, n1]) #membuat array kosong output1 dengan ukuran yang sama

for baris in range(0, m1-1): #melakukan iterasi untuk setiap piksel dalam citra copyCamera

    for kolom in range(0, n1-1):

        a1 = baris #menyimpan indeks baris saat ini dalam variabel a1

        b1 = kolom #menyimpan indeks kolom saat ini dalam variabel b1

        output1[a1, b1] = copyCamera[baris, kolom] \* 1.9 #menghitung nilai piksel baru dalam citra hasil perkalian konstanta (output1

fig, axes = plt.subplots(5, 2, figsize=(20, 20)) #membuat objek gambar dengan ukuran 5x2 dan Ukuran total gambar akan memiliki lebar 20 dan tinggi 20

ax = axes.ravel() #melakukan perataan (flattening) pada array axes untuk menghasilkan array satu dimensi ax

ax[0].imshow(image, cmap=plt.cm.gray) #menampilkan citra image

ax[0].set\_title("Citra Input") #memberi judul gambar

ax[1].hist(image.ravel(), bins=256) #menampilkan histogram dari citra image

ax[1].set\_title('Histogram Input') #memberi judul gambar

ax[2].imshow(image\_equalized, cmap=plt.cm.gray) #menampilkan citra image equalized

ax[2].set\_title("Citra Output HE") #memberi judul gambar

ax[3].hist(image\_equalized.ravel(), bins=256) #menampilkan histogram dari citra image\_equalized

ax[3].set\_title('Histogram Output HE Method') #memberi judul gambar

ax[4].imshow(image\_cs, cmap=plt.cm.gray) #menampilkan citra image cs

ax[4].set\_title("Citra Output CS") #memberi judul gambar

ax[5].hist(image\_cs.ravel(), bins=256) #menampilkan histogram dari citra image\_cs

ax[5].set\_title('Histogram Output CS Method')#memberi judul gambar

ax[6].imshow(image\_clahe, cmap=plt.cm.gray) #menampilkan citra image chale

ax[6].set\_title("Citra Grayscale CLAHE") #memberi judul gambar

ax[7].hist(image\_clahe.ravel(), bins=256) #menampilkan histogram dari citra image\_chale

ax[7].set\_title('Histogram Output CLAHE Method') #memberi judul gambar

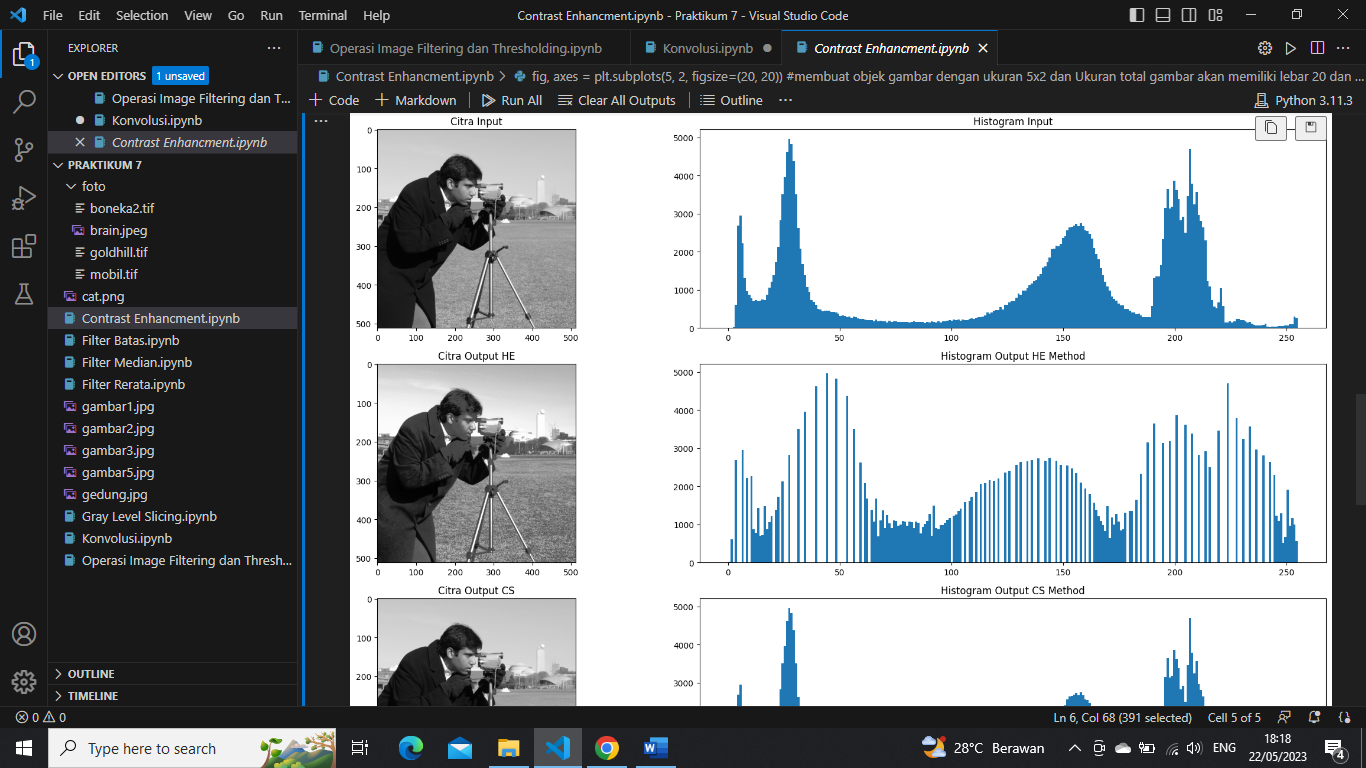
ax[8].imshow(output1, cmap=plt.cm.gray) #menampilkan citra output1

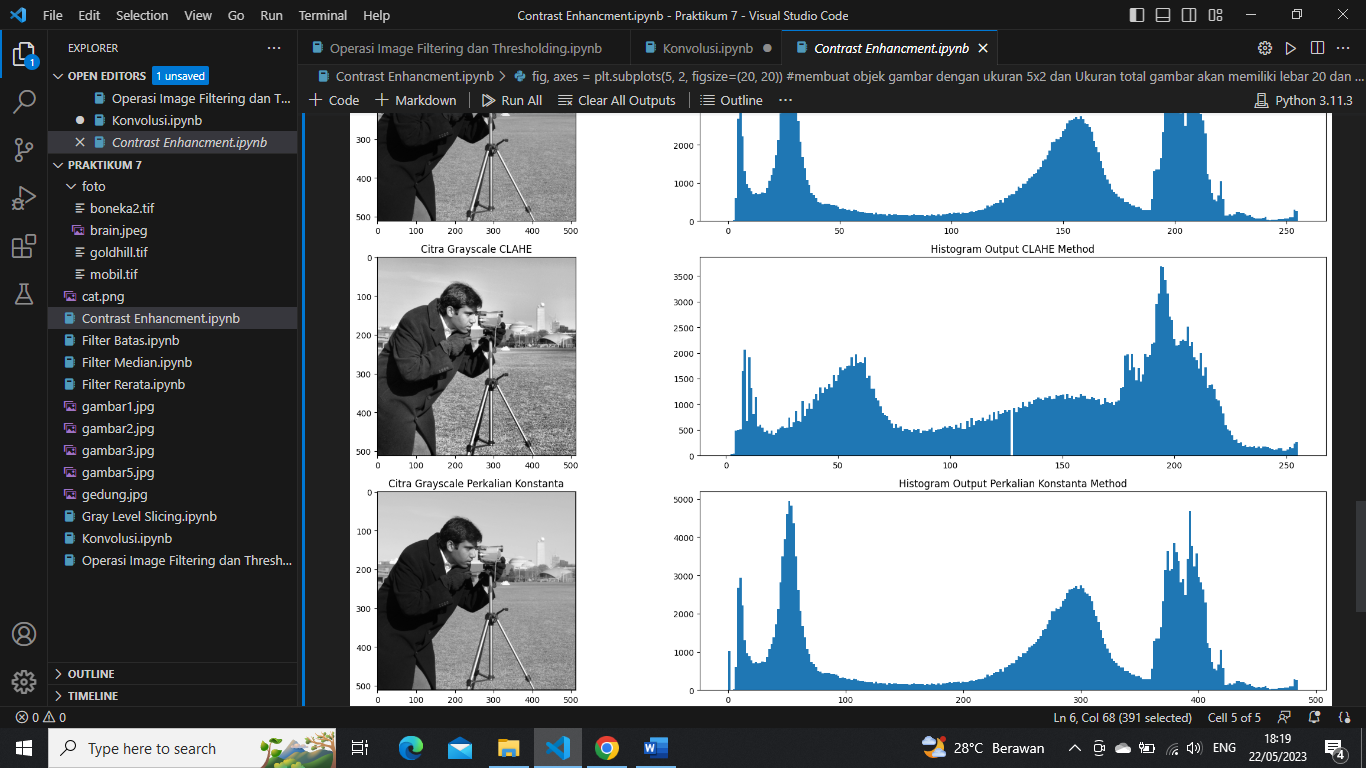
ax[8].set\_title("Citra Grayscale Perkalian Konstanta") #memberi judul gambar

ax[9].hist(output1.ravel(), bins=256) #menampilkan histogram dari citra output1

ax[9].set\_title('Histogram Output Perkalian Konstanta Method') #memberi judul gambar

fig.tight\_layout() #mengoptimalkan tata letak (layout) dari gambar-gambar yang ditampilkan dalam satu figure secara otomatis





**Gray Level Slicing**

import cv2 #mengimport modul cv2

import numpy as np #mengimport modul numpy

from skimage import data #mengimport modul skimage

import matplotlib.pyplot as plt #mengimport modul matplotlib

%matplotlib inline

img = cv2.imread('foto/elang.jpg',0) #membaca gambar

row, column = img.shape #mendapatkan dimensi gambar dengan memanggil metode .shape pada img

img1 = np.zeros((row,column),dtype = 'uint8') #membuat matriks nol dengan ukuran yang sama dengan gambar img

min\_range = 10 #menentukan rentang nilai minimum

max\_range = 60 #menentukan rentang nilai maksimum

for i in range(row): #melakukan iterasi untuk setiap piksel dalam gambar img

    for j in range(column):

        if img[i,j]>min\_range and img[i,j]<max\_range: #jika nilai piksel lebih besar dari min\_range dan lebih kecil dari max\_range maka :

            img1[i,j] = 255 #piksel di gambar img1 akan diatur menjadi 255 (putih)

        else: #jika tidak

            img1[i,j] = 0 #piksel di gambar img1 akan diatur menjadi 0 (hitam)

fig, axes = plt.subplots(2, 2, figsize=(7, 7)) #membuat objek subplot dengan ukuran 2x2 dan ukuran totalnya adalah 12x12

ax = axes.ravel() #meratakan matriks subplot menjadi array satu dimensi

ax[0].imshow(img, cmap=plt.cm.gray) #menampilkan citra input pada subplot indeks 0

ax[0].set\_title("Citra Input") #memberikan judul pada gambar

ax[1].hist(img.ravel(), bins=256) #menampilkan histogram dari citra input pada subplot indeks 1

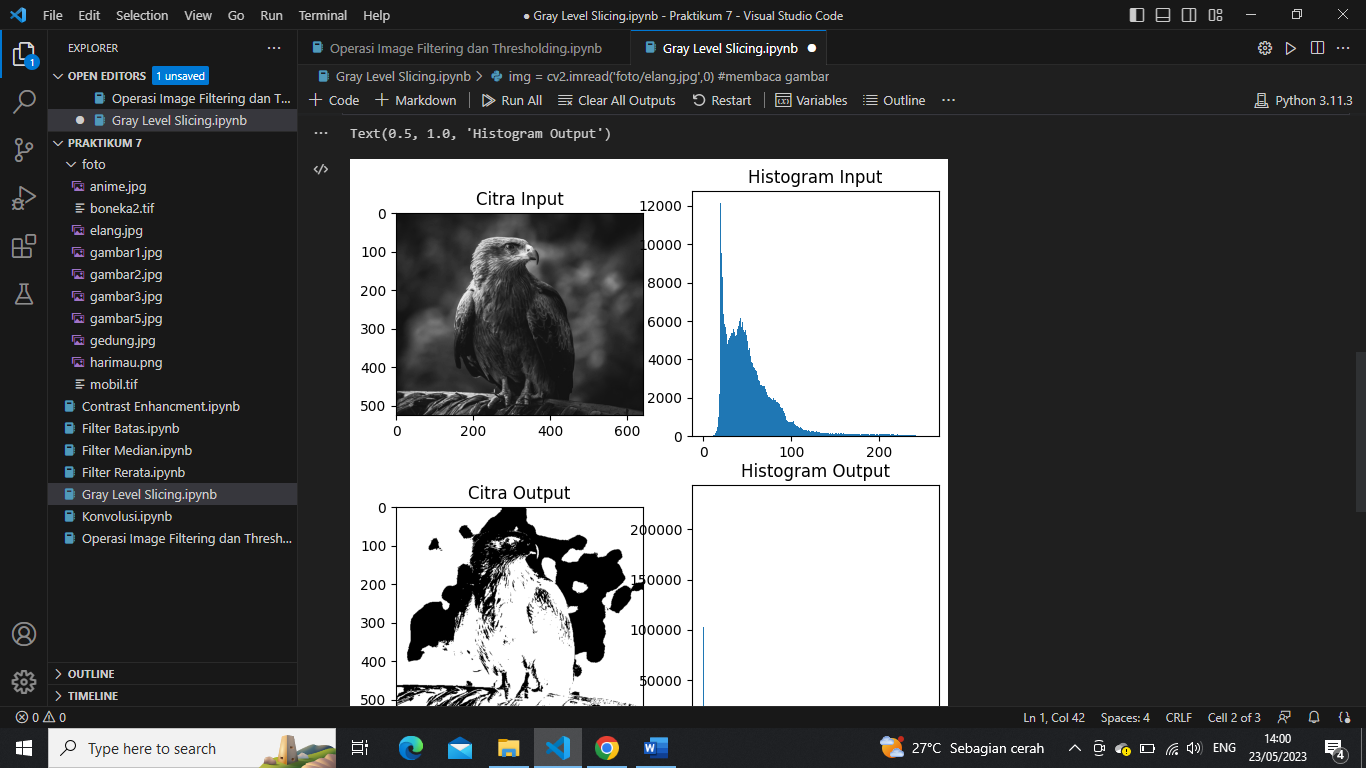
ax[1].set\_title('Histogram Input') #memberikan judul pada gambar

ax[2].imshow(img1, cmap=plt.cm.gray) #menampilkan citra input pada subplot indeks 2

ax[2].set\_title("Citra Output") #memberikan judul pada gambar

ax[3].hist(img1.ravel(), bins=256) #menampilkan histogram dari citra input pada subplot indeks 3

ax[3].set\_title('Histogram Output') #memberikan judul pada gambar



**Operasi Image Filtering dan Thresholding**

Low-Pass Filtering

import cv2 #mengimport modul cv2

import numpy as np #mengimport modul numpy

from matplotlib import pyplot as plt #mengimport modul matplotlib

image = cv2.imread('foto/elang.jpg') #membaca gambar 'cat.png'ndan menyimpannya dalam variabel img

elang = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB) #mengonversi gambar dari mode warna BGR ke mode warna RGB dan menyimpannya dalam variabel cat

cv2.imshow('foto/elang.jpg',image) #menampilkan gambar asli

kernel = np.ones((5, 5), np.float32) / 25 #membuat kernel 5x5 dengan nilai 1/25 untuk digunakan dalam proses filter

print(kernel) #mencetak kernel yang sudah dibuat

elang\_filter = cv2.filter2D(image,-1,kernel) #menerapkan filter 2D pada gambar img menggunakan kernel yang telah dibuat

fig, axes = plt.subplots(1, 2, figsize=(8, 8)) #membuat objek gambar dengan ukuran 5x2 dan Ukuran total gambar akan memiliki lebar 20 dan tinggi 20

ax = axes.ravel() #melakukan perataan (flattening) pada array axes untuk menghasilkan array satu dimensi ax

ax[0].imshow(elang\_filter, cmap=plt.cm.gray) #menampilkan citra image

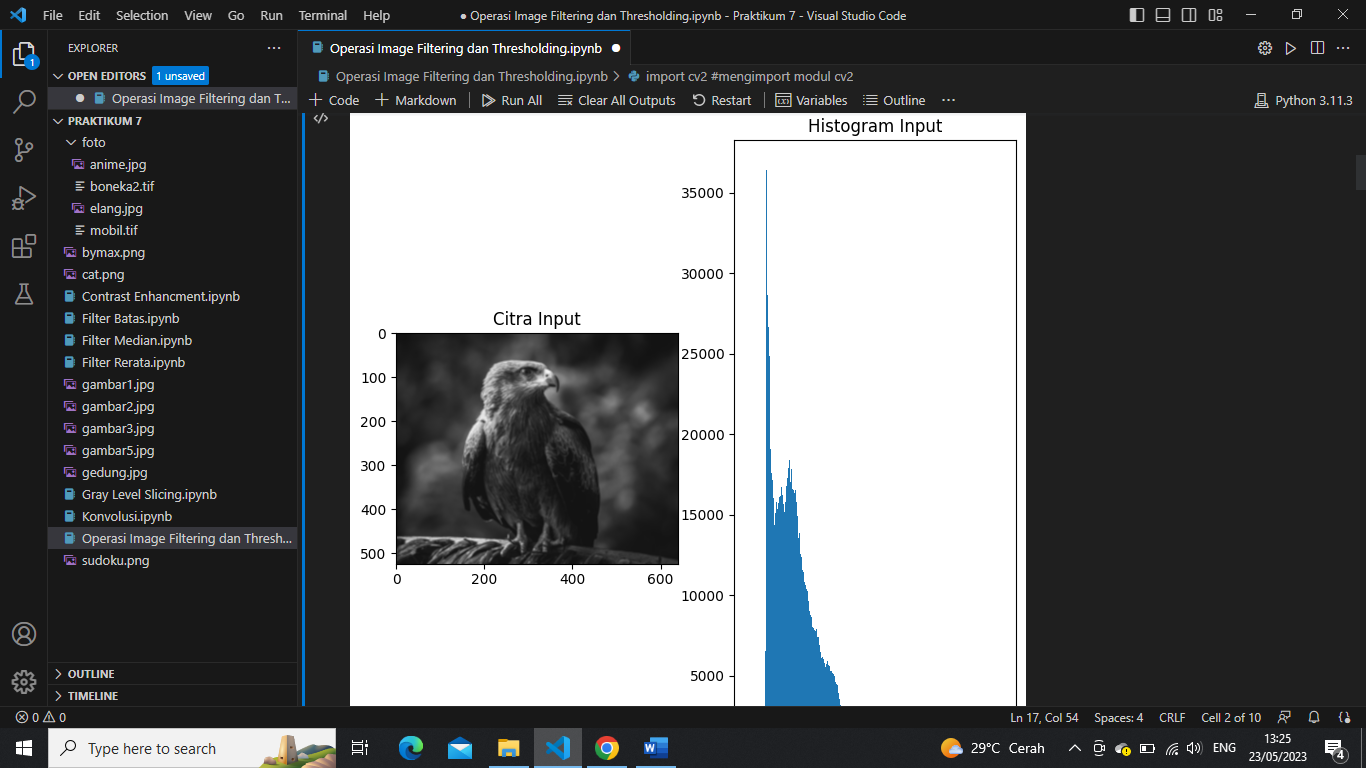
ax[0].set\_title("Citra Input") #memberi judul gambar

ax[1].hist(image.ravel(), bins=256) #menampilkan histogram dari citra image

ax[1].set\_title('Histogram Input') #memberi judul gambar

cv2.waitKey(0) #menunggu pengguna menekan tombol apa pun

cv2.destroyAllWindows() #menghapus semua jendela yang ditampilkan



plt.rcParams["figure.figsize"] = (7,7) #mengatur ukuran gambar plot menjadi 8x8

plt.subplot(2, 2, 1) #membuat subplot pertama dengan indeks 2, ,2, 1

plt.imshow(elang) #menampilkan gambar asli

plt.title('Original') #memberi judul

plt.xticks([]) #menghilangkan sumbu x

plt.yticks([]) #menghilangkan sumbu y

plt.subplot(2, 2, 2) #membuat subplot kedua dengan indeks 2, 2, 2

plt.imshow(elang\_filter) #menampilkan hasil filter

plt.title('Averaging') #memberi judul

plt.xticks([]) #menghilangkan sumbu x

plt.yticks([]) #menghilangkan sumbu y

plt.subplot(2, 2, 3) #membuat subplot kedua dengan indeks 2, 2, 3

plt.hist(elang.flatten(), 256, [0, 256]) #menampilkan gambar histogram dari gambar asli

plt.xlim([0, 256]) #untuk mengatur rentang sumbu x (nilai intensitas) dalam histogram

plt.title('Histogram Gambar Asli') #memberi judul

plt.subplot(2, 2, 4) #membuat subplot kedua dengan indeks 2, 2, 4

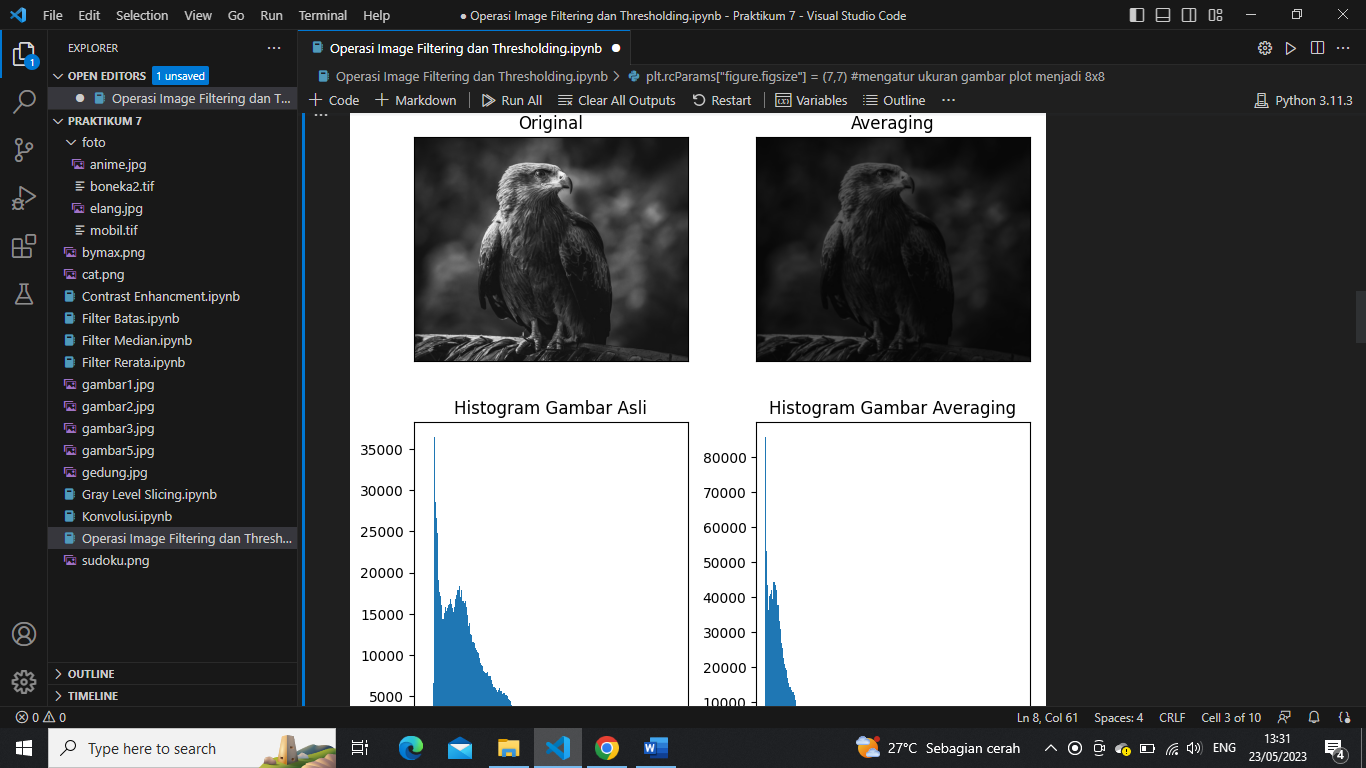
plt.hist(elang\_filter.flatten(), 256, [0, 256])

plt.xlim([0, 256]) #untuk mengatur rentang sumbu x (nilai intensitas) dalam histogram

plt.title('Histogram Gambar Averaging') #memberi judul

plt.tight\_layout() #mengoptimalkan tata letak (layout) dari gambar-gambar yang ditampilkan dalam satu figure secara otomatis

plt.show() #menampilkan gambar



kernel = np.matrix([ #membuat sebuah kernel dengan ukuran 3x3 yang disimpan dalam variabel kernel

          [1, 1, 1],

          [1, 2, 1],

          [1, 1, 1]

          ])/25

print(kernel)

elang\_filter = cv2.filter2D(image,-1,kernel) #membuat filter

fig, axes = plt.subplots(1, 2, figsize=(8, 8)) #membuat objek gambar dengan ukuran 5x2 dan Ukuran total gambar akan memiliki lebar 20 dan tinggi 20

ax = axes.ravel() #melakukan perataan (flattening) pada array axes untuk menghasilkan array satu dimensi ax

ax[0].imshow(elang\_filter, cmap=plt.cm.gray) #menampilkan citra image

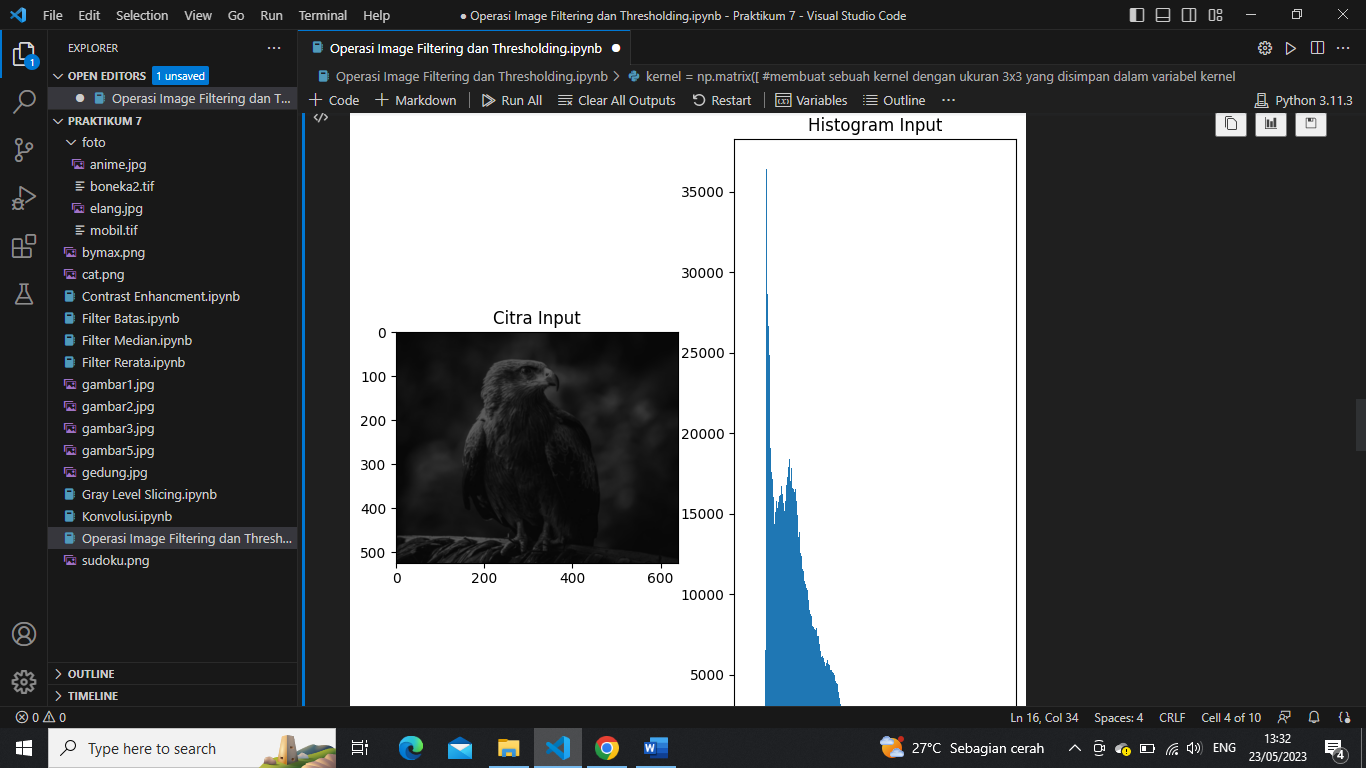
ax[0].set\_title("Citra Input") #memberi judul gambar

ax[1].hist(image.ravel(), bins=256) #menampilkan histogram dari citra image

ax[1].set\_title('Histogram Input') #memberi judul gambar

cv2.waitKey(0) #menunggu pengguna menekan tombol apa pun

cv2.destroyAllWindows() #menghapus semua jendela yang ditampilkan



High-Pass Filtering

import cv2 #mengimport modul cv2

import numpy as np #mengimport modul numpy

from matplotlib import pyplot as plt #mengimport modul matplotlib

img = cv2.imread('foto/harimau.png',0) #membaca gambar dalam mode grayscale dan menyimpan ke variabel img

laplacian = cv2.Laplacian(img,cv2.CV\_64F) #menerapkan operator Laplacian dan menyimpan hasilnya dalam variabel laplacian

sobelx = cv2.Sobel(img,cv2.CV\_64F,1,0,ksize=7) #menerapkan filter Sobel ke arah sumbu x dan ukuran kernel 5x5.

sobely = cv2.Sobel(img,cv2.CV\_64F,0,1,ksize=7) #menerapkan filter Sobel ke arah sumbu y dan ukuran kernel 5x5.

plt.rcParams["figure.figsize"] = (20,20) #mengatur ukuran gambar yang akan ditampilkan

plt.subplot(2,2,1),plt.imshow(img,cmap = 'gray') #membuat subplot dengan ukuran 2x2 dan menampilkan gambar asli pada subplot 1

plt.title('Original'), plt.xticks([]), plt.yticks([]) #memberikan judul dan menghilangkan sumbu x dan y

plt.subplot(2,2,2),plt.imshow(laplacian,cmap = 'gray') #membuat subplot pada indeks 2 dan menampilkan gambar hasil operator Laplacian

plt.title('Laplacian'), plt.xticks([]), plt.yticks([]) #memberikan judul dan menghilangkan sumbu x dan y

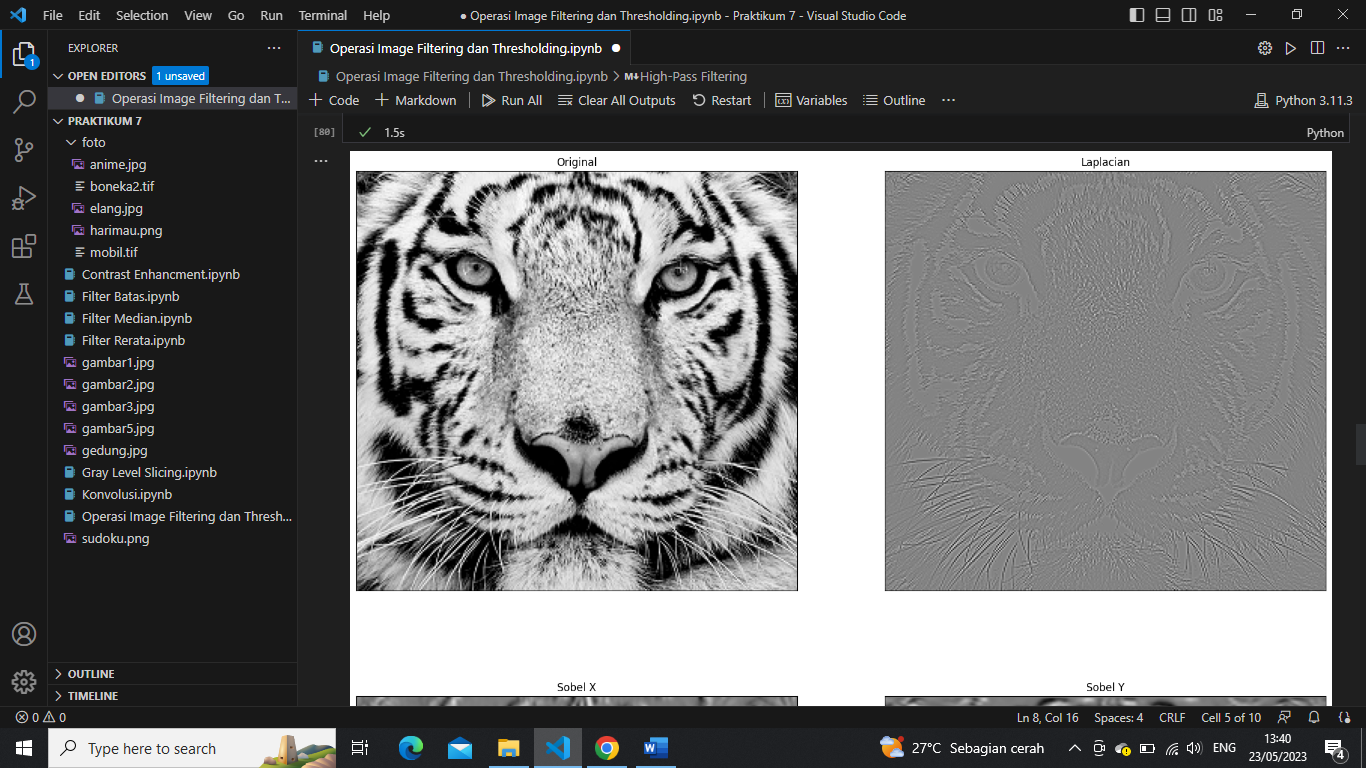
plt.subplot(2,2,3),plt.imshow(sobelx,cmap = 'gray') #membuat subplot pada indeks 3 dan menampilkan gambar hasil operator Sobel pada sumbu x

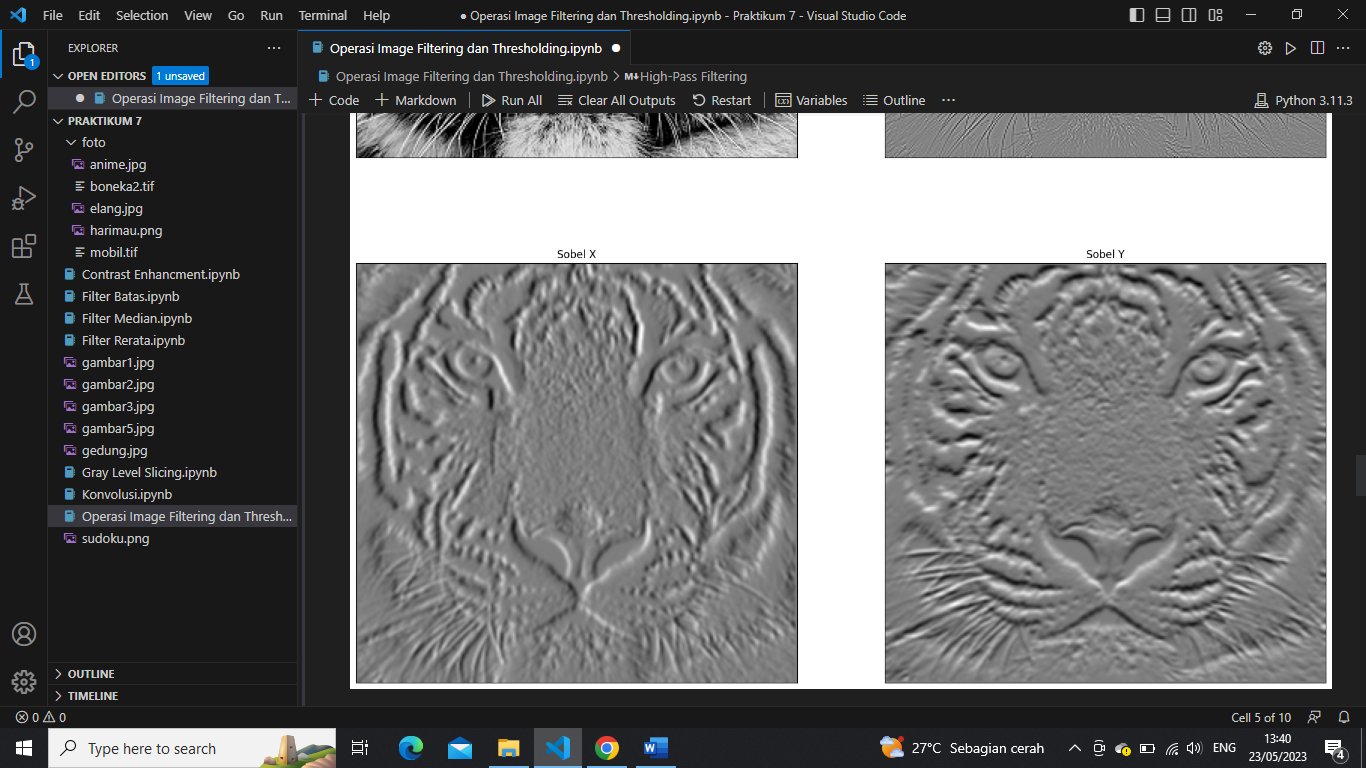
plt.title('Sobel X'), plt.xticks([]), plt.yticks([]) #memberikan judul dan menghilangkan sumbu x dan y

plt.subplot(2,2,4),plt.imshow(sobely,cmap = 'gray') #membuat subplot pada indeks 4 dan menampilkan gambar hasil operator Sobel pada sumbu y

plt.title('Sobel Y'), plt.xticks([]), plt.yticks([]) #memberikan judul dan menghilangkan sumbu x dan y

plt.show()





img = cv2.imread('foto/anime.jpg',0) #membaca gambar dalam mode grayscale dan menyimpan ke variabel img

edges = cv2.Canny(img,100,200) #menerapkan operator Canny pada gambar img, angka 100 dan 200 adalah nilai ambang bawah dan ambang atas untuk deteksi tepi

plt.subplot(3, 2, 1) #membuat subplot dengan 1 baris dan 2 kolom

plt.imshow(img,cmap = 'gray') #menampilkan gambar

plt.title('Original Image') #memberikan judul

plt.xticks([]) #menghilangkan tanda sumbu pada subplot

plt.yticks([]) #menghilangkan tanda sumbu pada subplot

plt.subplot(3, 2, 2) #memilih subplot kedua sebagai area tampilan

plt.imshow(edges,cmap = 'gray') #menampilkan gambar hasil deteksi tepi

plt.title('Edge Image') #memberikan judul

plt.xticks([]), plt.yticks([]) #menghilangkan tanda sumbu pada subplot

plt.subplot(3, 2, 3) #membuat subplot kedua dengan indeks 2, 2, 3

plt.hist(img.flatten(), 256, [0, 256]) #menampilkan gambar histogram dari gambar asli

plt.xlim([0, 256]) #untuk mengatur rentang sumbu x (nilai intensitas) dalam histogram

plt.title('Histogram Gambar Asli') #memberi judul

plt.subplot(3, 2, 4) #membuat subplot kedua dengan indeks 2, 2, 3

plt.hist(edges.flatten(), 256, [0, 256]) #menampilkan gambar histogram dari gambar asli

plt.xlim([0, 256]) #untuk mengatur rentang sumbu x (nilai intensitas) dalam histogram

plt.title('Histogram Gambar Asli') #memberi judul

plt.show() #menampilkan gambar

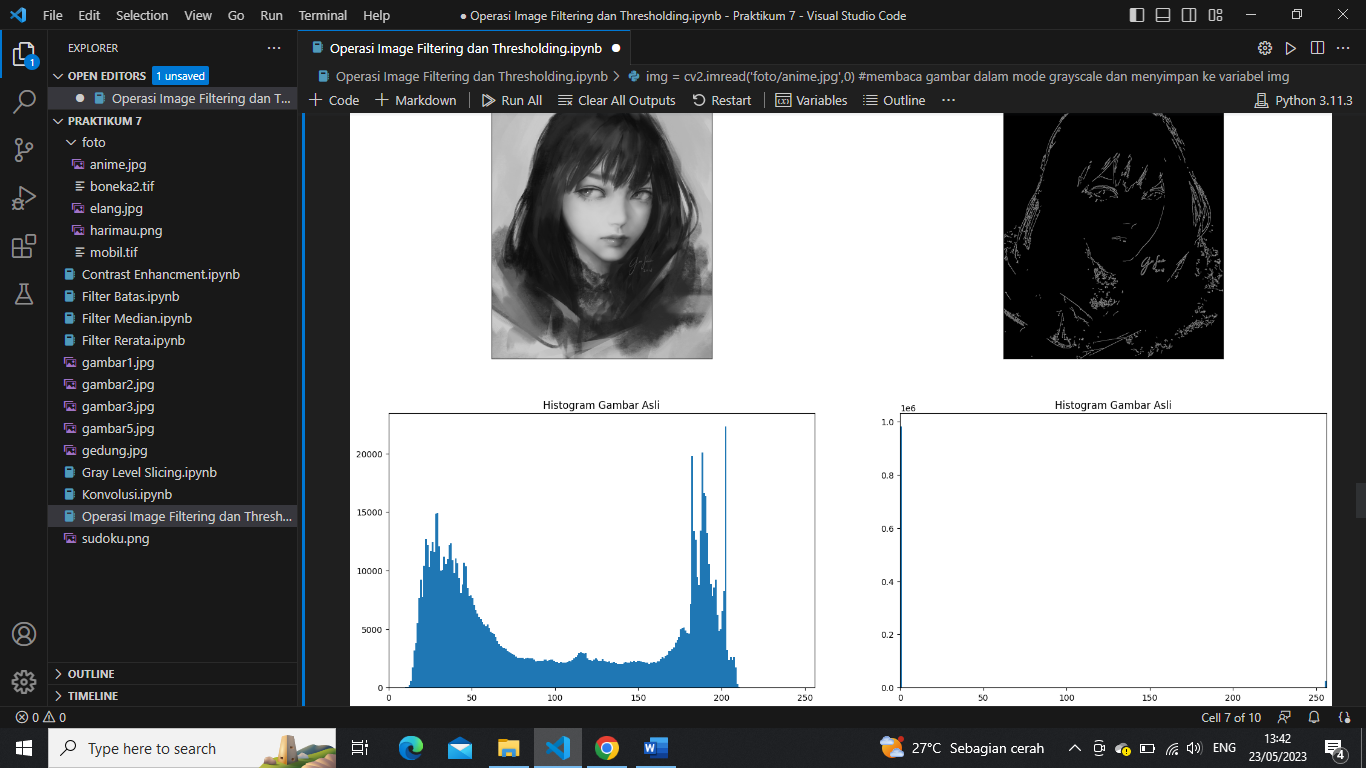


Image Thresholding

img = cv2.imread('foto/anime.jpg',0) #membaca gambar bymax

#Perhitungan Thresholding

ret,thresh1 = cv2.threshold(img,150,255,cv2.THRESH\_BINARY)

ret,thresh2 = cv2.threshold(img,150,255,cv2.THRESH\_BINARY\_INV)

ret,thresh3 = cv2.threshold(img,150,255,cv2.THRESH\_TRUNC)

ret,thresh4 = cv2.threshold(img,150,255,cv2.THRESH\_TOZERO)

ret,thresh5 = cv2.threshold(img,150,255,cv2.THRESH\_TOZERO\_INV)

titles = ['Gambar asli','BINARY','BINARY\_INV','TRUNC','TOZERO','TOZERO\_INV'] #membuat daftar judul untuk setiap gambar

images = [img, thresh1, thresh2, thresh3, thresh4, thresh5] #menampilkan daftar gambar yang akan ditampilkan

for i in range(6): #melakukan loop sebanyak 6 kali

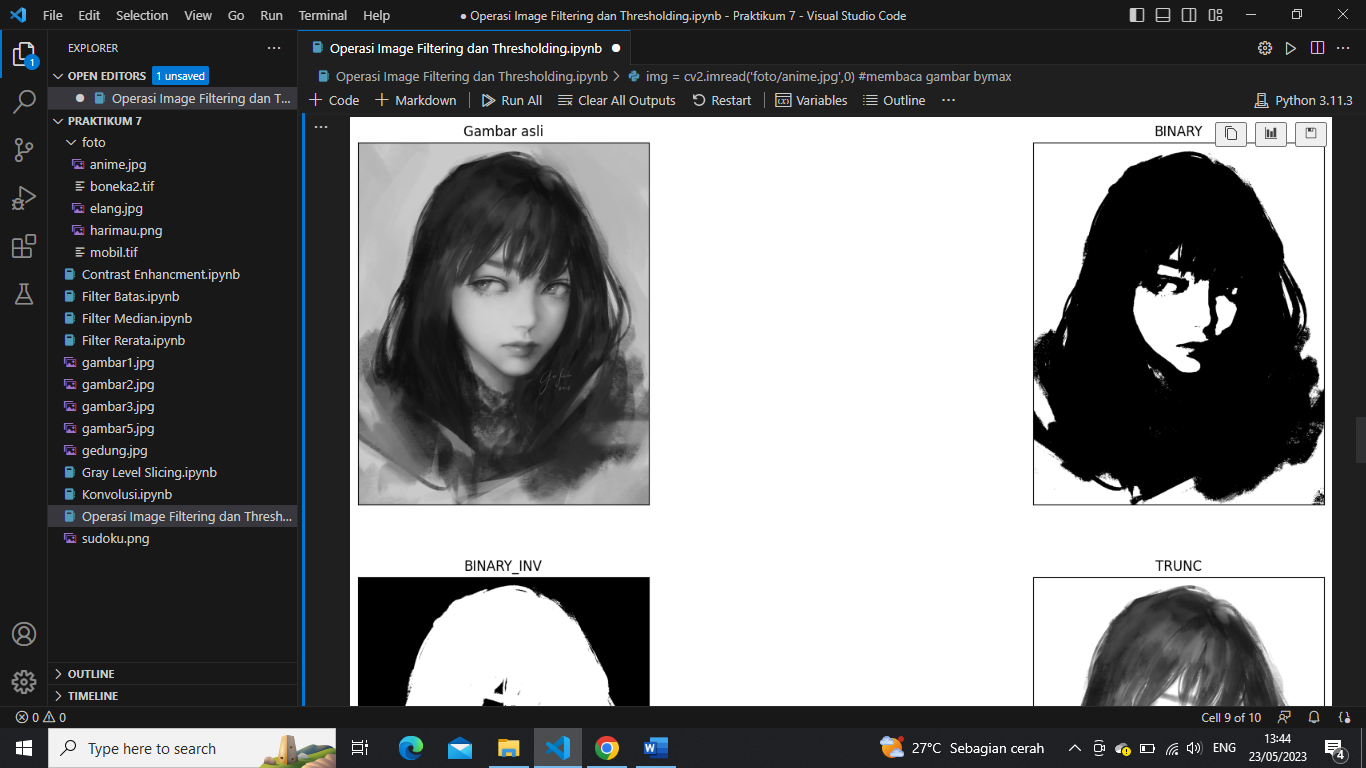
    plt.subplot(3,2,i+1) #untuk mengatur subplot dengan tata letak 3 baris dan 2 kolom

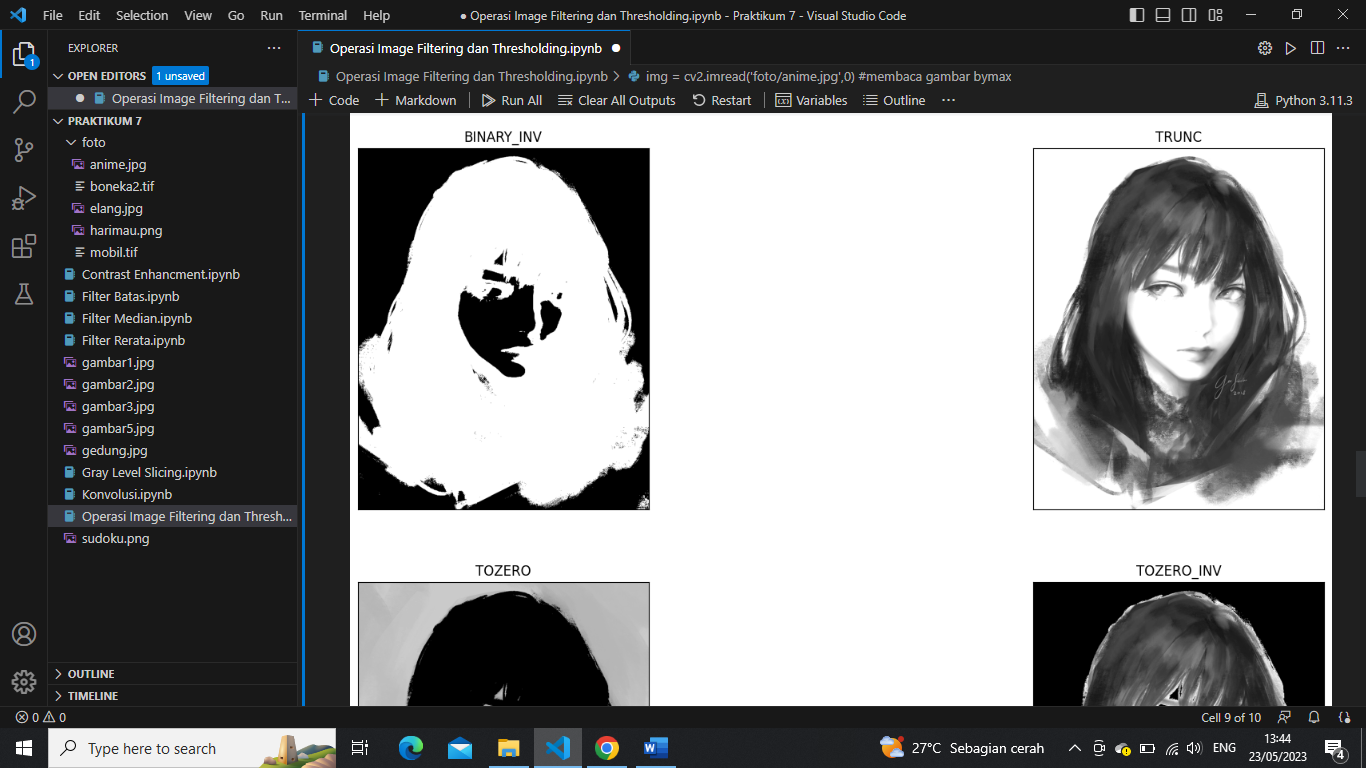
    plt.imshow(images[i],'gray')

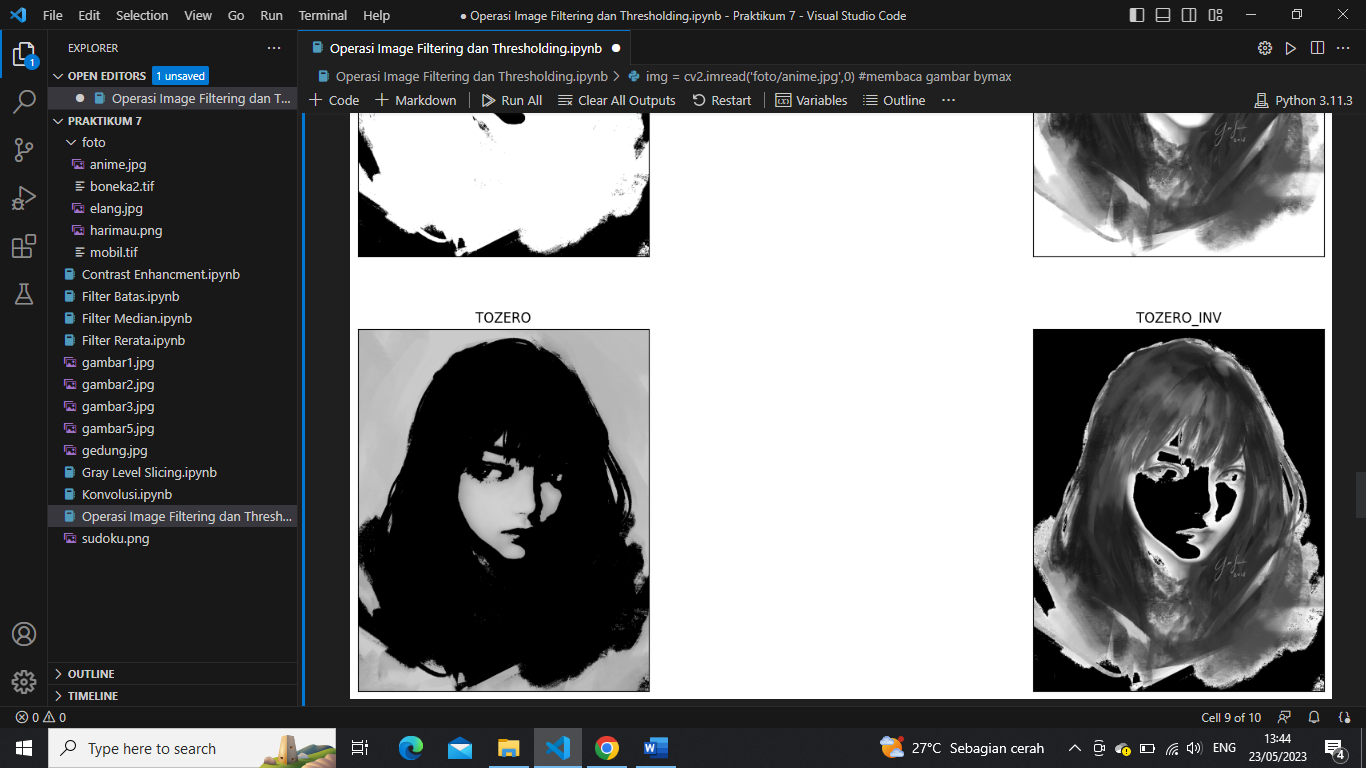
    plt.title(titles[i]) #mengatur judul pada setiap subplot

    plt.xticks([]),plt.yticks([]) #menghilangkan penanda sumbu x dan sumbu y pada setiap subplot

plt.show() #menampilkan gambar







img = cv2.medianBlur(img,5) #untuk menghaluskan tepi ojek pada gambar

ret,th1 = cv2.threshold(img,127,255,cv2.THRESH\_BINARY) #melakukan thresholding binary global dengan nilai ambang 127 dan disimpan ke variabel th1

th2 = cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE\_THRESH\_MEAN\_C,\

            cv2.THRESH\_BINARY,11,2) #melakukan thresholding adaptif menggunakan metode rata-rata dan disimpan dalam variabel th2

th3 = cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE\_THRESH\_GAUSSIAN\_C,\

            cv2.THRESH\_BINARY,11,2) #melakukan thresholding adaptif menggunakan metode Gaussian dan disimpan dalam variabel th3

titles = ['Original Image', 'Global Thresholding (v = 127)','Adaptive Mean Thresholding', 'Adaptive Gaussian Thresholding'] #membuat daftar judul untuk setiap gambar

images = [img, th1, th2, th3] #membuat daftar gambar yang akan ditampilkan pada subplot

for i in range(4): #melakukan loop sebanyak 4 kali

    plt.subplot(2,2,i+1) #untuk mengatur subplot dengan tata letak 3 baris dan 2 kolom

    plt.imshow(images[i],'gray')

    plt.title(titles[i]) #mengatur judul pada setiap subplot

    plt.xticks([]),plt.yticks([]) #menghilangkan penanda sumbu x dan sumbu y pada setiap subplot

plt.show() #menampilkan gambar

