Nama : Muhammad Baehaqi

NIM : 1207070134

Mat a Kuliah : Praktikum Pengolahan Citra Digital (TSEB)

Link GitHub : <https://github.com/MBaehaqi13/praktikum-morfologi-dan-cnn>

Google Colab : https://colab.research.google.com/drive/1tZnhzLNu3HR9jA7QnDVyXFeGoC3DcftQ

**MORFOLOGI**

**PERCOBAAN CLOSING**

**Soure Code :**

#import library

import cv2

import numpy as np

from skimage import data

from skimage.io import imread

import matplotlib.pyplot as plt

#import image

#image = data.retina()

#image = data.astronaut()

image = imread(fname="aqua2.jpg")

print(image.shape)

plt.imshow(image)

plt.show()

#PROSES MORFOLOGI CLOSING

hsv = cv2.cvtColor(image, cv2.COLOR\_BGR2HSV)

# defining the range of masking

blue1 = np.array([110, 50, 50])

blue2 = np.array([130, 255, 255])

# initializing the mask to be

# convoluted over input image

mask = cv2.inRange(hsv, blue1, blue2)

# passing the bitwise\_and over

# each pixel convoluted

res = cv2.bitwise\_and(image, image, mask = mask)

# defining the kernel i.e. Structuring element

kernel = np.ones((5, 5), np.uint8)

# defining the closing function

# over the image and structuring element

closing = cv2.morphologyEx(mask, cv2.MORPH\_CLOSE, kernel)

#PLOT CITRA OUTPUT

fig, axes = plt.subplots(1, 2, figsize=(12, 12))

ax = axes.ravel()

ax[0].imshow(mask)

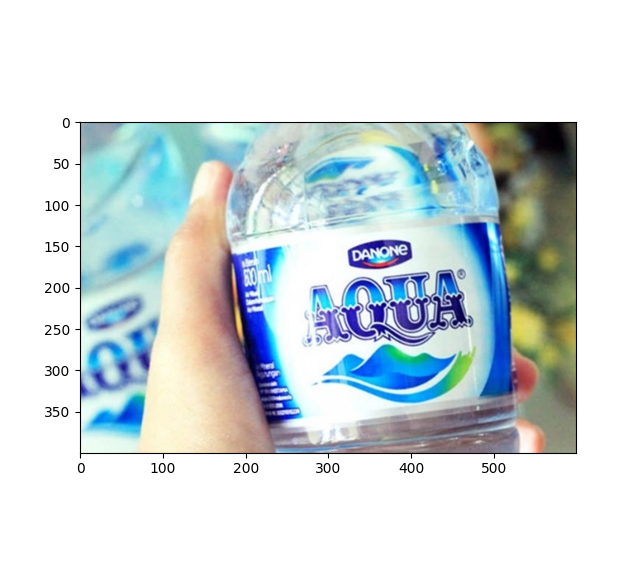
ax[0].set\_title("Citra Input 1")

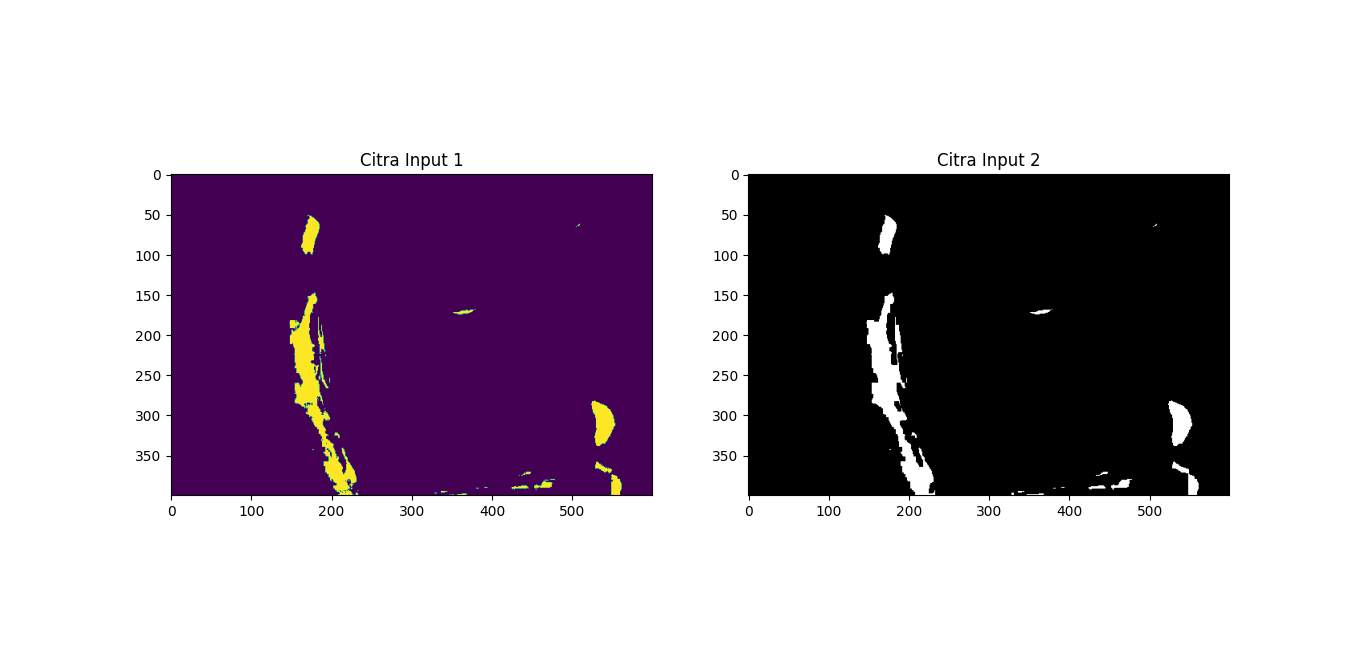
ax[1].imshow(closing, cmap='gray')

ax[1].set\_title('Citra Input 2')

plt.show()

**Hasil :**

****

****

**PERCOBAAN DILASI & EROSI**

**Source Code :**

#import library

import cv2

import numpy as np

import matplotlib.pyplot as plt

#import image

# Reading the input image

img = cv2.imread('saya.jpg', 0)

# Taking a matrix of size 5 as the kernel

kernel = np.ones((5,5), np.uint8)

#proses dilasi dan erosi

img\_erosion = cv2.erode(img, kernel, iterations=1)

img\_dilation = cv2.dilate(img, kernel, iterations=1)

#plot image

fig, axes = plt.subplots(3, 2, figsize=(20, 20))

ax = axes.ravel()

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

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

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

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

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

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

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

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

ax[4].imshow(img\_dilation, cmap = 'gray')

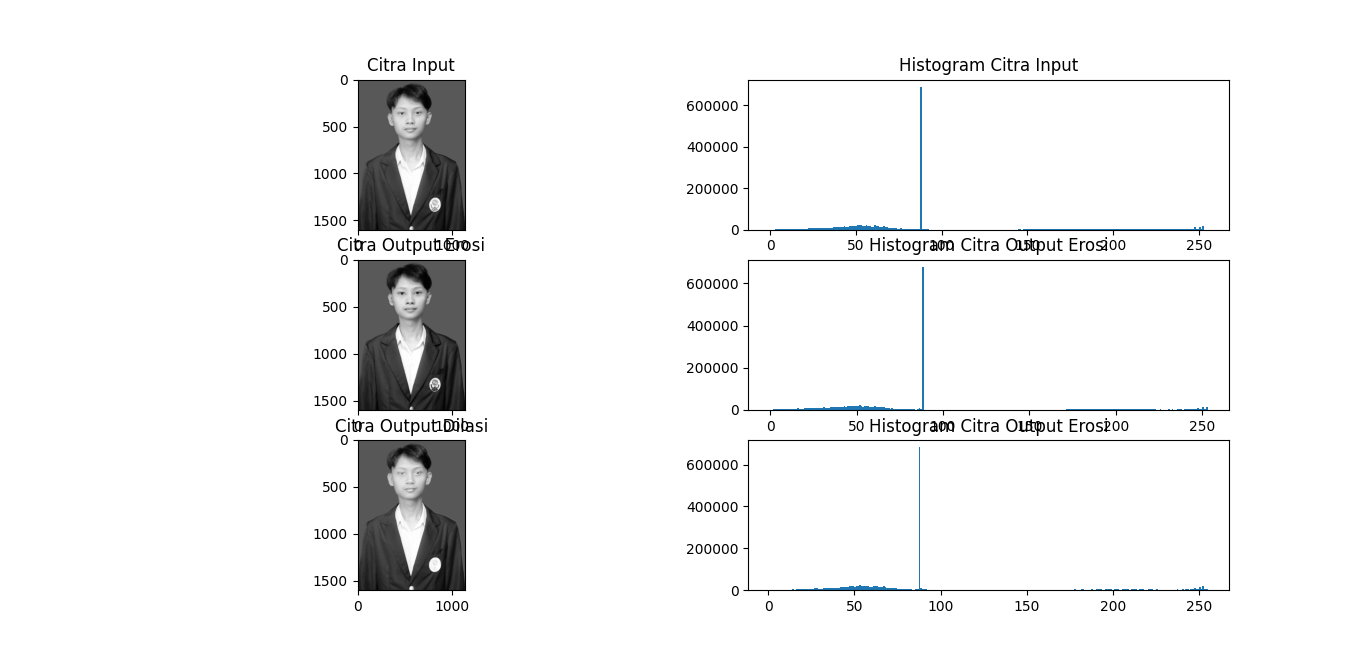
ax[4].set\_title("Citra Output Dilasi")

ax[5].hist(img\_dilation.ravel(), bins = 256)

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

plt.show()

**Hasil :**

****

**PERCOBAAN GRADIENT**

**Source Code :**

# Python programe to illustrate

# Gradient morphological operation

# on input frames

# organizing imports

import cv2

import numpy as np

# return video from the first webcam on your computer.

screenRead = cv2.VideoCapture(0)

# loop runs if capturing has been initialized.

while(1):

    # reads frames from a camera

    \_, image = screenRead.read()

    # Converts to HSV color space, OCV reads colors as BGR

    # frame is converted to hsv

    hsv = cv2.cvtColor(image, cv2.COLOR\_BGR2HSV)

    # defining the range of masking

    blue1 = np.array([110, 50, 50])

    blue2 = np.array([130, 255, 255])

    # initializing the mask to be

    # convoluted over input image

    mask = cv2.inRange(hsv, blue1, blue2)

    # passing the bitwise\_and over

    # each pixel convoluted

    res = cv2.bitwise\_and(image, image, mask = mask)

    # defining the kernel i.e. Structuring element

    kernel = np.ones((5, 5), np.uint8)

    # defining the gradient function

    # over the image and structuring element

    gradient = cv2.morphologyEx(mask, cv2.MORPH\_GRADIENT, kernel)

    # The mask and closing operation

    # is shown in the window

    cv2.imshow('Gradient', gradient)

    # Wait for 'a' key to stop the program

    if cv2.waitKey(1) & 0xFF == ord('a'):

        break

# De-allocate any associated memory usage

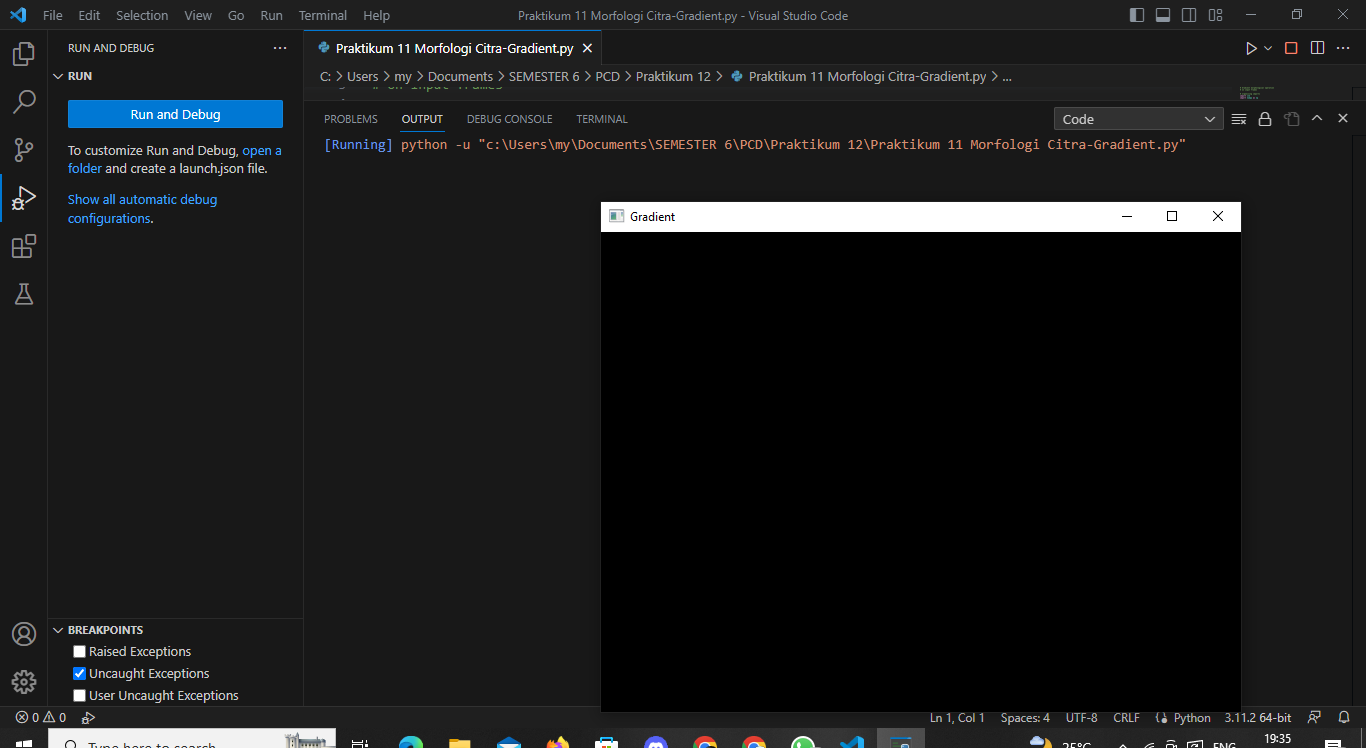
cv2.destroyAllWindows()

# Close the window / Release webcam

screenRead.release()

**Hasil :**

**( Device yang saya gunakan tidak memiliki Kamera web)**



**PERCOBAAN OPENING**

**Source Code :**

# Python programe to illustrate

# Closing morphological operation

# on an image

# organizing imports

import cv2

import numpy as np

# return video from the first webcam on your computer.

screenRead = cv2.VideoCapture(0)

# loop runs if capturing has been initialized.

while(1):

    # reads frames from a camera

    \_, image = screenRead.read()

    # Converts to HSV color space, OCV reads colors as BGR

    # frame is converted to hsv

    hsv = cv2.cvtColor(image, cv2.COLOR\_BGR2HSV)

    # defining the range of masking

    blue1 = np.array([110, 50, 50])

    blue2 = np.array([130, 255, 255])

    # initializing the mask to be

    # convoluted over input image

    mask = cv2.inRange(hsv, blue1, blue2)

    # passing the bitwise\_and over

    # each pixel convoluted

    res = cv2.bitwise\_and(image, image, mask = mask)

    # defining the kernel i.e. Structuring element

    kernel = np.ones((5, 5), np.uint8)

    # defining the closing function

    # over the image and structuring element

    closing = cv2.morphologyEx(mask, cv2.MORPH\_CLOSE, kernel)

    # The mask and closing operation

    # is shown in the window

    cv2.imshow('Mask', mask)

    cv2.imshow('Closing', closing)

    # Wait for 'a' key to stop the program

    if cv2.waitKey(1) & 0xFF == ord('a'):

        break

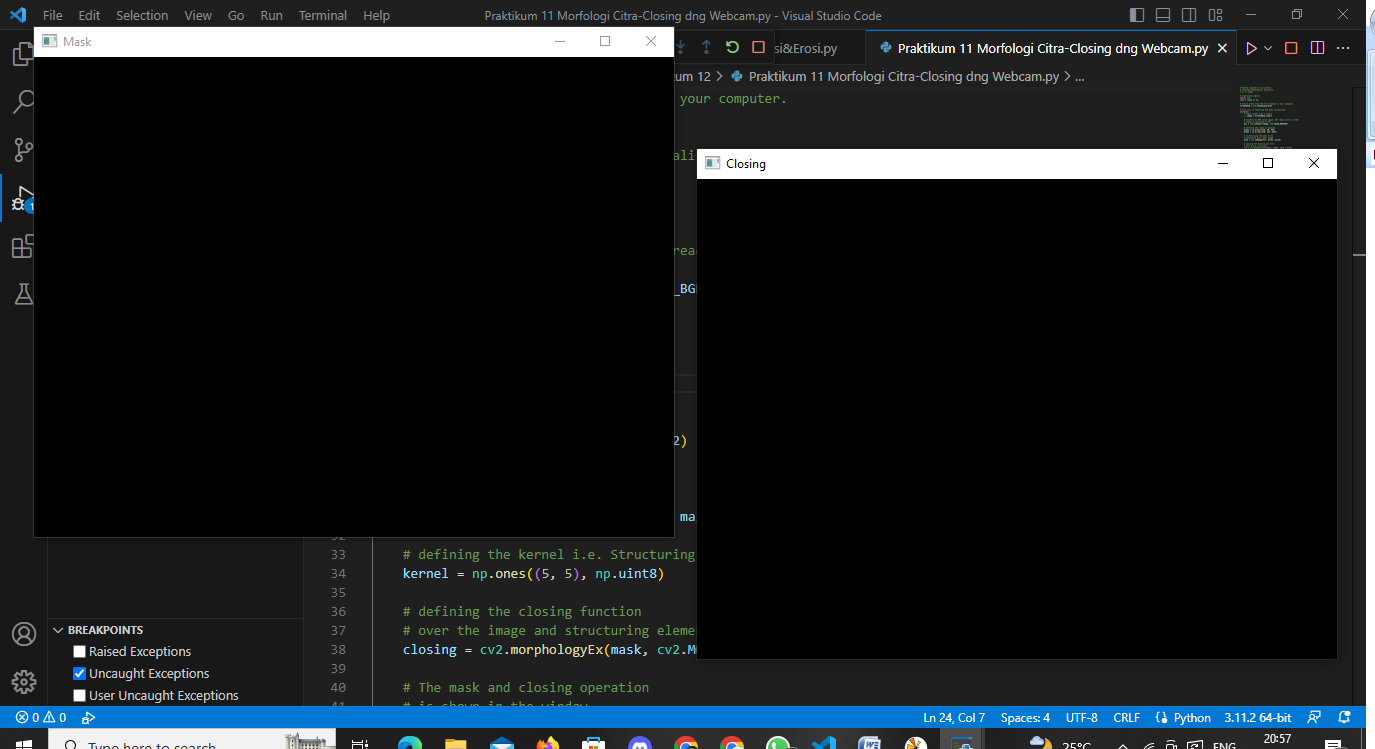
# De-allocate any associated memory usage

cv2.destroyAllWindows()

# Close the window / Release webcam

screenRead.release()

**Hasil :**



**Analisis :**

Percobaan closing, dilasi, erosi, gradient, dan opening merupakan bagian dari operasi morfologi pada citra digital. Operasi-operasi ini digunakan untuk memanipulasi bentuk dan struktur objek dalam citra. Berikut adalah analisis masing-masing operasi tersebut:

1. Closing:

Closing merupakan gabungan operasi dilasi diikuti dengan erosi pada citra digital. Tujuan dari operasi closing adalah untuk mengisi lubang kecil dan menyambung celah yang terputus dalam objek. Hal ini berguna untuk menghilangkan noise kecil dan menghasilkan objek yang lebih halus. Closing juga dapat memperbesar ukuran objek.

1. Dilasi:

Dilasi adalah operasi yang digunakan untuk memperbesar atau memperluas objek dalam citra. Pada setiap piksel dalam citra, dilasi mencari piksel tetangganya dan memperluas objek dengan menambahkan piksel tetangga tersebut ke dalam objek. Hal ini menyebabkan objek menjadi lebih besar dan lebih tebal.

1. Erosi:

Erosi adalah operasi yang digunakan untuk menyusutkan atau mengurangi ukuran objek dalam citra. Pada setiap piksel dalam citra, erosi mencari piksel tetangganya dan memeriksa apakah semua piksel tetangga tersebut termasuk dalam objek. Jika tidak, piksel tersebut dihapus dari objek. Erosi menghilangkan piksel-piksel tepi objek dan menyebabkan objek menjadi lebih kecil dan lebih tipis.

1. Gradient:

Gradient merupakan perbedaan antara citra hasil dilasi dan citra hasil erosi. Operasi ini digunakan untuk mengungkapkan tepi objek dalam citra. Dengan mengurangi citra erosi dari citra dilasi, piksel yang terletak pada tepi objek akan memiliki perbedaan nilai yang tinggi, sementara piksel di dalam objek akan memiliki perbedaan nilai yang rendah.

1. Opening:

Opening merupakan gabungan operasi erosi diikuti dengan dilasi pada citra digital. Tujuan dari operasi opening adalah untuk menghilangkan noise dan menghaluskan kontur objek dalam citra. Opening dapat mengurangi ukuran objek dan memisahkan objek yang saling menempel.

Secara keseluruhan, percobaan closing, dilasi, erosi, gradient, dan opening dapat digunakan dalam berbagai aplikasi pengolahan citra, seperti pemrosesan gambar medis, pengenalan pola, segmentasi objek, dan banyak lagi. Dengan memahami prinsip dan efek dari masing-masing operasi ini, Anda dapat mengoptimalkan analisis citra digital untuk mencapai tujuan yang diinginkan.