NAMA : Rifqi Alfinnur Charisma

import cv2

import numpy as np

import matplotlib.pyplot as plt

*# membaca citra covid dan citra mask*

img = cv2.imread('covid\_image.png')

img\_mask = cv2.imread('mask.png')

import library yang dibutuhkan lalu tampilkan citra x-ray dan citra maskingnya.

*# operasi histogram*

hist,bins = np.histogram(img.flatten(),256,[0,256])  *# histogram citra covid*

cdf = hist.cumsum() *# cdf = cumulative distribution function*

cdf\_m = np.ma.masked\_equal(cdf,0) *#masking*

cdf\_m = (cdf\_m - cdf\_m.min())\*255/(cdf\_m.max()-cdf\_m.min()) *#normalisasi*

cdf = np.ma.filled(cdf\_m,0).astype('uint8') *#mengembalikan ke dalam tipe data uint8*

img2 = cdf[img] *#mengubah citra menjadi citra yang sudah di normalisasi*

cdf\_normalized = cdf \* float(hist.max()) / cdf.max() *#normalisasi histogram*

plt.subplot(2, 1, 1)

plt.plot(cdf, *color* = 'b') *#plot histogram*

plt.hist(img.flatten(),256,[0,256], *color* = 'r') *# menampilkan histogram citra covid*

plt.xlim([0,256]) *#batas x*

plt.legend(('cdf','histogram'), *loc* = 'upper left') *# menampilkan keterangan historgram*

plt.show()

plt.subplot(2, 1, 2)

plt.plot(cdf\_normalized, *color* = 'b') *# menampilkan bingkai histogram*

plt.hist(img2.flatten(),256,[0,256], *color* = 'r') *# menampilkan histogram citra covid*

plt.xlim([0,256]) *#batas x*

plt.legend(('cdf','histogram'), *loc* = 'upper left') *# menampilkan keterangan historgram*

plt.show()

*# menampilkan dan menyimpan citra histogram awal dan akhir*

cv2.imshow('img1',img)

cv2.imshow('img2',img2)

cv2.imwrite('hist\_awal.png',img)

cv2.imwrite('hist\_akhir.png',img2)

cv2.waitKey(0)

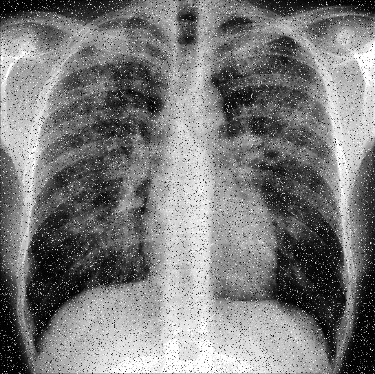
cv2.destroyAllWindows()

melakukan distribusi citra dengan menggunakan fungsi CDF, dimana nanti citra akan ditampilkan histogramnya, lalu algoritma akan memindai apakah citra low atau high sehingga sistem otomatis akan meratakan distribusi dari cita tersebut.

Histogram

Description automatically generated

Histogram yang sebelumnya mengumpul di tengah sekarang sudah terdistribusi secara merata

Citra juga sudah langsung diperbaiki oleh sistem sehingga lebih jelas.

*# operasi bitwise*

img = cv2.imread('hist\_akhir.png')

bitwise\_and = cv2.bitwise\_and(img, img\_mask, *mask*=None) *# operasi bitwise and*

*# menampilkan dan menyimpan citra masking akhir*

cv2.imshow('image', bitwise\_and)

cv2.imwrite('res\_bitwise.png', bitwise\_and)

cv2.waitKey(0)

cv2.destroyAllWindows()

selanjutnya citra akan dimasking hingga menampilkan gambar paru-parunya. Disini saya menggunakan operasi bitwise karena bitwise bersifat menumpuk citra satu dengan yang lainnya.



Citra sudah menampilkan paru-parunya saja.

*# salt and pepper removal*

img = cv2.imread('res\_bitwise.png')

kernel\_3\_3 = np.ones((3,3),np.float32)/9 *# kernel 3x3*

img\_snp\_average\_filter = cv2.filter2D(img,cv2.CV\_8U,kernel\_3\_3,(-1,-1), *delta* = 0, *borderType* = cv2.BORDER\_DEFAULT) *# operasi average filter*

img\_snp\_median\_median = cv2.medianBlur(img,3) *# operasi median filter*

*# menampilkan dan menyimpan citra yang sudah di filter*

cv2.imshow('img s&p reduction with average filter', img\_snp\_average\_filter)

cv2.imshow('img s&p reduction with median filter', img\_snp\_median\_median)

cv2.imwrite('res\_noise\_removal.png', img\_snp\_median\_median)

cv2.waitKey(0)

cv2.destroyAllWindows()

Masih terdapat noise pada citra sehingga kita perlu memperbakinya. Disini saya menggunakan operasi average filter dan fungsi median filter untuk mencari filter manakah yang paling baik untuk menghilangkan noise, setelah dilakukan percobaan maka fungsi median filter lebih cocok untuk kasus ini.



Hasil citra yang sudah dihilangkan noisenya.

*# operasi threshold*

img = cv2.imread('res\_noise\_removal.png')

ret,thresh1 = cv2.threshold(img,127,255,cv2.THRESH\_BINARY) *# threshold binary => 127*

ret,thresh2 = cv2.threshold(img,127,255,cv2.THRESH\_BINARY\_INV) *# threshold binary inverse => 127*

ret,thresh3 = cv2.threshold(img,127,255,cv2.THRESH\_TRUNC) *# threshold truncate => 127*

ret,thresh4 = cv2.threshold(img,127,255,cv2.THRESH\_TOZERO) *# threshold to zero => 127*

ret,thresh5 = cv2.threshold(img,127,255,cv2.THRESH\_TOZERO\_INV) *# threshold to zero inverse => 127*

titles = ['Original Image','BINARY','BINARY\_INV','TRUNC','TOZERO','TOZERO\_INV'] *# menampilkan kategori*

images = [img, thresh1, thresh2, thresh3, thresh4, thresh5] *# menampilkan gambar*

*# looping untuk menampilkan gambar + kategori*

for i in range(6):

    plt.subplot(2,3,i+1),plt.imshow(images[i],'gray')

    plt.title(titles[i])

    plt.xticks([]),plt.yticks([])

plt.show()

Karena tim medis hanya menginginkan bagian yang terinfeksi saja maka perlu dilakukannya thresholding. Di sini saya mencoba 5 fungsi threshold dan menset nilai threshold sebesar 127. Untuk hasil sebagai berikut :

A picture containing text

Description automatically generated

Bisa dilihat bahwa binary threshold lebih pas untuk kasus ini karena menampilkan jelas bagian yang terinfeksi saja.

*# menampilkan dan menyimpan citra final*

cv2.imshow('img', thresh1)

cv2.imwrite('final.png', thresh1)

cv2.waitKey(0)

cv2.destroyAllWindows()

Setelah itu kita simpan hasil final dari pemrosesan citra. Untuk hasil finalnya sebagai berikut :

