|  |  |  |
| --- | --- | --- |
|  | **Lembar Kerja Mahasiswa**  **Mata Kuliah Pengolahan Citra Digital Praktik (203311-20)**  **Program Studi Informatika**  **Fakultas Sains & Teknologi – Universitas Teknologi Yogyakarta** | |
|  | **Identitas Mahasiswa** | |
| **Nama** | **Arieska Restu Harpian Dwika** |
| **NPM** | **5200411488** |
| **Kelompok Prak** | **Kel. I** |
|  | | |
| **Soal 1.** | | |
| Berdasarkan demo tentang masking citra menggunakan bitwise operator di kelas, buatlah aplikasi berbasis GUI untuk menjalankan code tersebut dengan citra input:  dan  Sehingga menghasilkan output    Anda perlu memperhatikan:   1. Citra yang anda dapatkan di elearning adalah citra asli sehingga perlu dilakukan resize terlebih dahulu. Proses resize dilakukan diluar aplikasi GUI yang dibuat. 2. Citra pesawat perlu di-threshold dengan nilai tertentu. Agar memudahkan mencari nilai terbaik, tambahkan komponen GUI untuk bisa mengubah nilai threshold secara dinamis (seperti lembar kerja binerisasi citra) 3. Anda dapat membuat layout GUI Anda sendiri. Buatlah layout GUI yang menarik dan tetap mudah digunakan. | | |
| **Hasil Script** | | |
| **//tuliskan script python Anda di sini**  import cv2  import numpy as np  import os  from tkinter import \*  from tkinter import font  from tkinter import filedialog  from ttkbootstrap import Style  from tkinter import ttk  import tkinter as tk  from PIL import Image, ImageTk  def setOriginal1(*img*):      imgTk = ImageTk.PhotoImage(*img*)      lblOriImg1.configure(*image*=imgTk)      lblOriImg1.image = imgTk      lblOriImg1.pack()  def setOriginal2(*img*):      imgTk = ImageTk.PhotoImage(*img*)      lblOriImg2.configure(*image*=imgTk)      lblOriImg2.image = imgTk      lblOriImg2.pack()  def setResult(*img*):      imgTk = ImageTk.PhotoImage(*img*)      lblResultImg.configure(*image*=imgTk)      lblResultImg.image = imgTk      lblResultImg.pack()  def opencv2Pill(*img*):  *img* = cv2.cvtColor(*img*, cv2.COLOR\_BGR2RGB)      imgPill = Image.fromarray(*img*)      return imgPill    def resizeImg(*img*, *width*, *height*):  *img* = cv2.resize(*img*, (*width*, *height*), *interpolation*=cv2.INTER\_CUBIC)      return *img*  def clipping(*intensity*):      if *intensity* < 0:          return 0      if *intensity* > 255:          return 255      return *intensity*  def browseImage1():      global fln1      fln1 = filedialog.askopenfilename(*initialdir*=os.getcwd(), *title*="Select Image File",  *filetypes*=(                                          ("All Files", "\*.\*",),                                          ("PNG File", "\*.png"),                                          ("JPG File", "\*.jpg"))                                      )        img = opencv2Pill(resizeImg(cv2.imread(fln1), 316, 210))      setOriginal1(img)  def browseImage2():      global fln2      fln2 = filedialog.askopenfilename(*initialdir*=os.getcwd(), *title*="Select Image File",  *filetypes*=(                                          ("All Files", "\*.\*",),                                          ("PNG File", "\*.png"),                                          ("JPG File", "\*.jpg"))                                      )      img = opencv2Pill(resizeImg(cv2.imread(fln2), 384, 256))      setOriginal2(img)  def masking():      global fln1, fln2, thresh      img1 = resizeImg(cv2.imread(fln1), 158, 105)      img2 = resizeImg(cv2.imread(fln2), 576, 384)        img1Shape = img1.shape      roi = img2[0:img1Shape[0], 0:img1Shape[1]]      img2gray = cv2.cvtColor(img1, cv2.COLOR\_BGR2GRAY)      # thresh = int(sldThresh.get())      ret, mask = cv2.threshold(img2gray, thresh, 255, cv2.THRESH\_BINARY)      maskInv = cv2.bitwise\_not(mask)        img2Bg = cv2.bitwise\_and(roi, roi, *mask*=mask)      img1Fg = cv2.bitwise\_and(img1, img1, *mask*=maskInv)      dst = cv2.add(img2Bg, img1Fg)      img2[0:img1Shape[0], 0:img1Shape[1]] = dst        setResult(opencv2Pill(img2))  def sldThreshMove(*e*):      global thresh        thresh = int(sldThresh.get())      masking()        lblValue.configure(*text*=f'Value of threshold : {thresh}')    if \_\_name\_\_ == '\_\_main\_\_':      fln1, fln2 = None, None      thresh = 147        style = Style()      window = style.master      frm = ttk.Frame(window, *style*='primary.TFrame')      frm.pack\_propagate(0)      frm.pack(*fill*=tk.BOTH, *expand*=1)      frmImgOri = ttk.Frame(frm, *style*='secondary.TFrame', *width*=900, *height*=500)      frmImgOri.grid(*row*=0, *column*=0, *padx*=20, *pady*=(20,0))      frmImgOri1 = ttk.Frame(frmImgOri, *style*='info.TFrame', *width*=316, *height*=210)      frmImgOri1.pack(*side*="left", *padx*=20, *pady*=30)      frmBtn = ttk.Frame(frmImgOri, *style*='secondary.TFrame', *width*=100, *height*=200)      frmBtn.pack(*side*="left", *padx*=20, *pady*=30)      frmImgOri2 = ttk.Frame(frmImgOri, *style*='info.TFrame', *width*=384, *height*=256)      frmImgOri2.pack(*side*="left", *padx*=20, *pady*=20)      frmSld = ttk.Frame(frm, *style*='secondary.TFrame', *width*=942, *height*=50)      frmSld.pack\_propagate(0)      frmSld.grid(*row*=1, *column*=0, *padx*=25, *pady*=(0,10))      frmImgRes = ttk.Frame(frm, *style*='secondary.TFrame', *width*=576, *height*=384)      frmImgRes.grid(*row*=2, *column*=0, *padx*=20, *pady*=20)      frmImgResult = ttk.Frame(frmImgRes, *style*='info.TFrame', *width*=576, *height*=384)      frmImgResult.pack\_propagate(0)      frmImgResult.grid(*row*=0, *column*=0, *padx*=20, *pady*=20)      btnBrowse1 = ttk.Button(frmBtn, *text*='Browse Image 1', *style*='info.TButton', *cursor*="hand2", *width*=14, *command*=browseImage1)      btnBrowse1.pack(*side*='top', *pady*=10)      btnBrowse2 = ttk.Button(frmBtn, *text*='Browse Image 2', *style*='info.TButton', *cursor*="hand2", *width*=14, *command*=browseImage2)      btnBrowse2.pack(*side*='top', *pady*=10)      btnMerger = ttk.Button(frmBtn, *text*='→', *style*='success.TButton', *cursor*="hand2", *width*=2, *command*=masking)      btnMerger.pack(*side*='top', *pady*=10)      btnExit = ttk.Button(frmBtn, *text*='Exit', *style*='danger.TButton', *cursor*="hand2", *command*=lambda: exit())      btnExit.pack(*side*='top', *pady*=10)      lblOriImg1 = ttk.Label(frmImgOri1)      lblOriImg2 = ttk.Label(frmImgOri2)      lblResultImg = ttk.Label(frmImgResult)        sldThresh = ttk.Scale(frmSld, *from\_*=-255, *to*=255, *value*=0, *orient*='horizontal', *style*='info.Horizontal.TScale', *length*=511, *command*=sldThreshMove)      lblValue = ttk.Label(frmSld, *text*=f'Value of threshold : {thresh}', *style*='info.Inverse.TLabel')        lblValue.pack(*side*='left', *padx*=50)      sldThresh.pack(*side*='left', *padx*=50, *pady*=0)      window.title("Masking - 5200411488")      # window.geometry("1280x720")      # window.resizable(0, 0)      window.mainloop() | | |
| **Hasil Running Aplikasi** | | |
| **//paste-kan tampilan aplikasi Anda di sini** | | |