Lampiran 1 Source Code Program

```
import glob
import cv2
import numpy as np
from PIL import Image

#fist dance
tari1="Nelayan"
data1="Tari"+tari1
data_ori1="ori/"+data1
temp_crop1="temporary/temp_crop/"+data1
temp_bi1="temporary/temp_binary/"+data1
temp_crop_max1="temporary/temp_crop_max/"+data1
data_pre="preprosesing/"+data1

black =  [0,0,0]
im = Image.open(data_ori1 + "/"+ tari1 + " 100.jpg")
width, height = im.size

# Crop the center of the original image into 300 x 350 pixel
new_width=300
new_height=350

#crop image into 300 x 352
left = round((width - new_width)/2)
top = round((height - new_height)/2)
x_right = round(width - new_width) - left
x_bottom = round(height - new_height) - top
right = width - x_right
bottom = height - x_bottom
new_bottom=right-new_width

#resize croped image into 300 x 350
def Reformat_Image(ImageFilePath,outputPath):

    from PIL import Image
    image = Image.open(ImageFilePath, 'r')
    width_set = 300
    height_set = 350
    image_size = image.size
    width = image_size[0]
    height = image_size[1]

    background = Image.new('RGB', (width_set, height_set),
(255, 255, 255, 255))
    offset_width = (int(round(((width_set - width) / 2),
0)), int(round(((height_set-height)),0)))
    background.paste(image, offset_width)
    background.save(outputPath)
```

```
i=0
# loop over the image paths
for imagePath in glob.glob(data_ori1 + "/*.jpg"):
      i+=1
    #Croping black color on left and right side
      im1 = cv2.imread(imagePath)
      cropped_image1 = im1[top:new_height, new_bottom:right]
      cv2.imwrite(temp_crop1+"/"+str(i)+".jpg",
cropped_image1)

    #Convert image to binary with treshold value 55
      img1 = cv2.imread(temp_crop1+"/"+str(i)+".jpg",0)
      ret, binary_img1 = cv2.threshold(img1, 55, 255,
cv2.THRESH_BINARY)
      cv2.imwrite(temp_bi1+"/"+str(i)+".jpg", binary_img1 )

    #Bounding Box the dancer (black color) and Crop it
      image1 = cv2.imread(temp_bi1+"/"+str(i)+".jpg")
      X1,Y1 = np.where(np.all(image1==black,axis=2))
      zipped1 = np.column_stack((Y1,X1))
      min_axis1=np.amin(zipped1,axis=0)
      max_axis1=np.amax(zipped1,axis=0)

X_croped_start1=min_axis1[0]
      Y_croped_start1=min_axis1[1]
      X_croped_end1=max_axis1[0]
      Y_croped_end1=max_axis1[1]
      width_croped_img1=X_croped_end1 - X_croped_start1
      height_croped_img1=Y_croped_end1 - Y_croped_start1
      x1, y1 = X_croped_start1, Y_croped_start1

      img_crop_input1=
cv2.imread(temp_crop1+"/"+str(i)+".jpg")
      crop_img_result1=
img_crop_input1[y1:y1+height_croped_img1,
x1:x1+width_croped_img1]
      cv2.imwrite(temp_crop_max1+"/"+str(i)+".jpg",
crop_img_result1 )


    #Resize Image into 300 x 350 (add backgound in left, top
right side)
      ImageFilePath = temp_crop_max1+"/"+str(i)+".jpg"
      outputPath=data_pre+"/"+str(i)+".jpg"
      Reformat_Image(ImageFilePath,outputPath)
```

```
import glob
import cv2
import numpy as np
from plantcv import plantcv as pcv
from skimage import img_as_bool, io, color, morphology,
img_as_ubyte
from skimage.util import invert
from skimage.morphology import skeletonize
import skimage.io
import warnings

warnings.filterwarnings("ignore",category=UserWarning)
#fist dance
tari1="Wiranjaya"
data1="Tari"+tari1
dataset="preprosesing/"+data1
databiner1="temporary/skl_temp_binary/"+data1
databinerfil1="temporary/skl_temp_binary_filling/"+data1
datahasil="temporary/skl_lee/"+data1
datasklpr1="skeleton/"+data1
pcv.params.debug = "none"
kernel = np.ones((10,10),np.uint8)

i=0
# loop over the image paths
for imagePath in sorted(glob.glob(dataset + "/*.jpg"),
key=len):
      i+=1
      img = cv2.imread(imagePath,0)
      blur = cv2.GaussianBlur(img,(5,5),0)
      ret1, th1 = cv2.threshold(blur, 65, 255,
cv2.THRESH_BINARY)
      cv2.imwrite(databiner1 +"/"+str(i)+".jpg", th1)

    #filling hole
      img_fill1 = cv2.morphologyEx(th1, cv2.MORPH_OPEN,
kernel)
      cv2.imwrite(databinerfil1 +"/"+str(i)+".jpg",
img_fill1)

      #skeleton = skeletonize(image)
      img = invert(img_fill1)
      image = img_as_bool(color.rgb2gray(img))
      skeleton = skeletonize(image, method='lee')
      skimage.io.imsave(datahasil +"/"+str(i)+".jpg",
img_as_ubyte(skeleton))

    #pruned of skeleton branch
      skl= cv2.imread(datahasil +"/"+str(i)+".jpg",0)
      ret, bw_img = cv2.threshold(skl, 127, 255,
cv2.THRESH_BINARY)
      pruned_skeleton, segmented_img, segment_objects =
pcv.morphology.prune(skel_img=bw_img, size=30)
      cv2.imwrite(datasklpr1+"/"+str(i)+".jpg",
pruned_skeleton)
```

```
from scipy.spatial import distance as dist
import matplotlib.pyplot as plt
import glob
import cv2
import re

dataset="skeleton/TariNelayan"
dataset2="skeleton/TariWiranjaya"
tari1=dataset[dataset.rfind("/")+1:]
tari2=dataset2[dataset2.rfind("/")+1:]

dir_hasil="Nelayan_Wiranjaya"
plot="hasil/"+dir_hasil+"/plot/"
hasil="hasil/"+dir_hasil+"/"

dirfiletext=dir_hasil+".txt"
file1 = open(dirfiletext, 'w')

index1 = {}
images = {}
index2 = {}
images2 = {}
gerakan=0
jump=0
sama=False
namaplot=""
hist_akhir=""
sudah_gerakan=False
result_print=""
sudah_print = False
gerakan_print=0
no_gerakan=0
total_jump=0
gambar=0
selisih_sebelumnya=0
selisih_sebelumnya2=0
n_awal=""
m_awal=""
m_awal_next=""
m_new=""
frame_gerakan =5
euc_dist=1.20

hog = cv2.HOGDescriptor()
hog2 = cv2.HOGDescriptor()

# loop over the fist image paths
for imagePath in sorted(glob.glob(dataset + "/*.jpg"),
key=len):
      filename = imagePath[imagePath.rfind("\\") + 1:]
      image = cv2.imread(imagePath)
```

```
# loop over the fist image paths
for imagePath in sorted(glob.glob(dataset + "/*.jpg"),
key=len):
      filename = imagePath[imagePath.rfind("\\") + 1:]
      image = cv2.imread(imagePath)
      images[filename] = image
      hist = hog.compute(image)
      hist = cv2.normalize(hist, hist).flatten()
      index1[filename] = hist

# loop over the second image paths
for imagePath2 in sorted(glob.glob(dataset2 + "/*.jpg"),
key=len):
      filename2 = imagePath2[imagePath2.rfind("\\") + 1:]
      image2 = cv2.imread(imagePath2)
      images2[filename2] = image2
      hist2 = hog2.compute(image2)
      hist2 = cv2.normalize(hist2, hist2).flatten()
      index2[filename2] = hist2

for (n, hist) in index1.items():

    ada_m_awal=""
    result_print=""

  #lompati bagian frame yg sudah terdeteksi sebagai sambungan
dari gerakan yg sudah berhasil ditemukan, agar ini tidak
dihitung lagi

    if ((total_jump >= frame_gerakan) & (jump>1)):
      jump-=1
      print(n)
      continue

    for (m, hist2) in index2.items():
      if (m_new != ""):
            m=m_new
            hist2=index2[m]
            m_new=""
        # compute the distance between the two histograms
      key_1=list(index1.keys())[-1]
      key_2=list(index2.keys())[-1]
      key_22=list(index2.keys())[-2]
      frame_min=frame_gerakan-0
      key_min3=list(index2.keys())[-3]
      key_min4=list(index2.keys())[-4]
      key_min5=list(index2.keys())[-5]
      key_min6=list(index2.keys())[-6]
      key_min7=list(index2.keys())[-7]

      d = dist.euclidean(hist, hist2)

      if (d <= euc_dist):
            cetak1=
```

```
      if (d <= euc_dist):
            cetak1=
":"+tari1+"/"+n+"&"+tari2+"/"+m+"="+str(round(d,2))+";"

          #mencari index terakhir
            name1=re.sub('[^0-9]','',n)
            name1=int(name1)
            name2=re.sub('[^0-9]','',m)
            name2=int(name2)

            key_first1=list(index1.keys())[0]
            key_first2=list(index2.keys())[0]


bk1=list(index1.keys())[list(index1.keys()).index(n) - 1]

bk2=list(index2.keys())[list(index2.keys()).index(m) - 1]

            name1_before=re.sub('[^0-9]','',bk1)
            name1b=int(name1_before)
            selisih_sebelumnya=name1-name1b

            name2_before=re.sub('[^0-9]','',bk2)
            name2b=int(name2_before)
            selisih_sebelumnya2=name2-name2b


 #jika sebelumnya sama = false (dan sekarang true) berarti
ini awal  gerakan
 #meskipun sebelumnya sama=True ada kemungkinan ini adalah
gerakan baru jika selisih dengan frame sebelumnya > 1
            if ((sama is False) | (selisih_sebelumnya != 1))
:
                #sama = false menandakan bahwa ini adalah
awal permulaan (frame pertama) dari gerakan baru
                n_awal=n
                hist_awal=hist
                m_awal=m
                hist_m_awal=hist2
                if (m_awal != key_2):

mk=list(index2.keys())[list(index2.keys()).index(m_awal) +
1]
                    m_awal_next=mk
                    hist_m_awal_next=index2[m_awal_next]

            if sudah_gerakan is False :

                    if (gambar >= frame_gerakan):
                        sudah_print=False
                    gambar=0
                    result_print=""
```

```
            if ((n != key_first1) & (m != key_first2)) :

                #jika frame ini adalah sambungan (selisih
dengan frame sebelumnya adalah 1) maka increment var jump
                if ((selisih_sebelumnya == 1) &
(selisih_sebelumnya2 == 1) & ((ada_m_awal=="") |
(ada_m_awal==1)) & (sama == True)) :

                    if (ada_m_awal == "") :
                        ada_m_awal=1

  #jika listnya sudah terahir, jangan di+ lagi
            if ((n != key_1)&(m != key_2)&(m != key_22)) :
                #cek nama file tari ke1 dan index
berikutnya jika selisihnya = 1 maka ini adalah sambungan
nk1=list(index1.keys())[list(index1.keys()).index(n) + 1]
                name1_next=re.sub('[^0-9]','',nk1)
                name1_next=int(name1_next)
                selisih1=name1_next-name1
nk2=list(index2.keys())[list(index2.keys()).index(m) + 1]
                name2_next=re.sub('[^0-9]','',nk2)
                name2_next=int(name2_next)
                selisih2=name2_next-name2

                #hanya jika tari 1 dan tari 2 adalah
sambungan baru kemudian dinaikkan index tari 1
                if ((selisih1 == 1) & (selisih2 == 1)):
nk=list(index1.keys())[list(index1.keys()).index(n) + 1]
                    n=nk
                    hist=index1[n]
mk=list(index2.keys())[list(index2.keys()).index(m) + 1]
                    m_new=mk
                    gambar+=1
                    sudah_gerakan=False
                elif (sama == True):
                    if (gambar >= frame_gerakan):
                        sudah_print=False
                    gambar=0
                    result_print=""
                    sudah_gerakan=True
                else :

                    sudah_gerakan=False
            else :
                sudah_gerakan=False
                sama=False
                break

            sama=True
            gerakan_print=gerakan+1;
```

```
distance_print=str(gerakan_print)+"."+str(gambar)+cetak1
            result_print=result_print+distance_print
          #jika distance >= nilai ambang
      else:
            #jika frame yg berurutan >=5 baru cetak output
           if ((gambar >= frame_gerakan)):
                gerakan+=1
                total_jump=gambar
                jump=gambar

                result_line = result_print.split(";")
                for result_item in result_line :
                    print (result_item)
                    if(result_item):

no_gerakan=result_item.split(":",1)
no_gerakan_full=no_gerakan[0]
no_gerakan_bl=no_gerakan_full.split(".",1)
no_gerakan_bl_print=no_gerakan_bl[1]
no_gerakan_awal_print=no_gerakan_bl[0]
no_gerakan_print=str(no_gerakan_awal_print)+"."+no_gerakan
_bl_print
                        namaplot=str(no_gerakan_print) +
".jpg"

file_tari1=result_item.split('/')[1].split('&')
                        nama_tari1=file_tari1[0]

file_tari2=result_item.split('/')[2].split('=')
                        nama_tari2=file_tari2[0]
                        distance_print=file_tari2[1]
                        file_text_item=result_item+"\n"
                        file1.writelines(file_text_item)
                        plotdir=plot+namaplot

                        #plot
                        fig_pic = plt.figure()
                        fig_pic.suptitle("%s:%s/%s &
%s/%s=%s"%
(no_gerakan_print,tari1,nama_tari1,tari2,nama_tari2,distan
ce_print), fontsize = 20)
                        ax = fig_pic.add_subplot(1, 2, 1)
                        ax.imshow(images[nama_tari1])
                        plt.axis("off")
                        ax = fig_pic.add_subplot(1, 2, 2)
                        ax.imshow(images2[nama_tari2])
                        plt.axis("off")
                        plt.savefig(plotdir,
bbox_inches="tight",pad_inches = 0.1)
                        plt.close(fig_pic)

                        #copy gambar ke folder yg terpisah
                        imageCopy =
images[nama_tari1].copy()
```

```
namaplot=str(no_gerakan_print) + ".jpg"
file_tari1=result_item.split('/')[1].split('&')
                        nama_tari1=file_tari1[0]
file_tari2=result_item.split('/')[2].split('=')
                        nama_tari2=file_tari2[0]
distance_print=file_tari2[1]
file_text_item=result_item+"\n"
file1.writelines(file_text_item)
                        plotdir=plot+namaplot
                        #plot
                        fig_pic = plt.figure()
                        fig_pic.suptitle("%s:%s/%s &
%s/%s=%s"%
(no_gerakan_print,tari1,nama_tari1,tari2,nama_tari2,distanc
e_print), fontsize = 20)
                        ax = fig_pic.add_subplot(1, 2, 1)
                        ax.imshow(images[nama_tari1])
                        plt.axis("off")
                        ax = fig_pic.add_subplot(1, 2, 2)
                        ax.imshow(images2[nama_tari2])
                        plt.axis("off")
                        plt.savefig(plotdir,
bbox_inches="tight",pad_inches = 0.1)
                        plt.close(fig_pic)

#copy gambar ke folder yg terpisah
                        imageCopy =
images[nama_tari1].copy()
cv2.imwrite(hasil+tari1+"/"+nama_tari1,imageCopy)
                        imageCopy2 =
images2[nama_tari2].copy()
cv2.imwrite(hasil+tari2+"/"+nama_tari2,imageCopy2)
                sudah_print=True
            if (ada_m_awal != ""):
                ada_m_awal=2
#jika gerakan yg ditemukan sudah berakhir (ditandai dengan
TRUE menjadi False) maka kursor kembali ke frame awal
gerakan pada video 1 (n_awal) untuk melanjutkan pencarian
ke sisa video 2 yang belum dicek
            if (( sama is True ) & (m != key_2) & (m !=
key_22) & (m != key_min3) & (m != key_min4) & (m !=
key_min5)& (m != key_min6) & (m != key_min7)):
                n=n_awal
                hist = hist_awal
                while (gambar > 1):
m_new=list(index2.keys())[list(index2.keys()).index(m) - 0]
                    gambar-=1
            sama=False
            gambar=0
cetak_jumlah=['Total banyaknya gerakan adalah
',str(no_gerakan_awal_print)]
print('Total banyaknya gerakan adalah
',no_gerakan_awal_print)
file1.writelines(cetak_jumlah)
file1.close()
```