**NAME: Manasi Saiba**

**REG NO:20178030(Information Technology)**

**Assignment-10(Image processing)**

WAP to enhance the given image using the following Segmentation Techniques:

1. **Thresholding**

import cv2

import numpy as np

image1 = cv2.imread('input1.jpg')

image2 = cv2.cvtColor(image1, cv2.COLOR\_BGR2GRAY)

ret, thresh1 = cv2.threshold(img, 120, 255, cv2.THRESH\_BINARY)

ret, thresh2 = cv2.threshold(img, 120, 255, cv2.THRESH\_BINARY\_INV)

ret, thresh3 = cv2.threshold(img, 120, 255, cv2.THRESH\_TRUNC)

ret, thresh4 = cv2.threshold(img, 120, 255, cv2.THRESH\_TOZERO)

ret, thresh5 = cv2.threshold(img, 120, 255, cv2.THRESH\_TOZERO\_INV)

cv2.imshow('Binary Threshold', thresh1)

cv2.imshow('Binary Threshold Inverted', thresh2)

cv2.imshow('Truncated Threshold', thresh3)

cv2.imshow('Set to 0', thresh4)

cv2.imshow('Set to 0 Inverted', thresh5)

if cv2.waitKey(0) & 0xff == 27:

    cv2.destroyAllWindows()

1. **Region-growing**

def get8n(x, y, shape):

out = []

if y-1 > 0 and x-1 > 0:

out.append( (y-1, x-1) )

if y-1 > 0 :

out.append( (y-1, x))

if y-1 > 0 and x+1 < shape[1]:

out.append( (y-1, x+1))

if x-1 > 0:

out.append( (y, x-1))

if x+1 < shape[1]:

out.append( (y, x+1))

if y+1 < shape[0] and x-1 > 0:

out.append( ( y+1, x-1))

if y+1 < shape[0] :

out.append( (y+1, x))

if y+1 < shape[0] and x+1 < shape[1]:

out.append( (y+1, x+1))

return out

def region\_growing(img, seed):

list = []

outimg = np.zeros\_like(img)

list.append((seed[0], seed[1]))

while(len(list)):

pix = list[0]

outimg[pix[0], pix[1]] = 255

for coord in get8n(pix[0], pix[1], img.shape):

if img[coord[0], coord[1]] > 0:

outimg[coord[0], coord[1]] = 255

list.append((coord[0], coord[1]))

list.pop(0)

return outimg

def on\_mouse(event, x, y, flags, params):

if event == cv2.EVENT\_LBUTTONDOWN:

print 'Seed: ' + str(x) + ', ' + str(y)

clicks.append((y,x))

clicks = []

image = cv2.imread('lena.jpg', 0)

ret, img = cv2.threshold(image, 200, 255, cv2.THRESH\_BINARY)

cv2.namedWindow('Input')

cv2.setMouseCallback('Input', on\_mouse, 0, )

cv2.imshow('Input', img)

cv2.waitKey()

seed = clicks[-1]

cv2.imshow('Region Growing', region\_growing(img, seed))

cv2.waitKey()

cv2.destroyAllWindows()

1. **K-mean clustering**

import numpy as np

import matplotlib.pyplot as plt

import cv2

%matplotlib inline

image = cv2.imread('images/monarch.jpg')

image = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB)

plt.imshow(image)

pixel\_vals = image.reshape((-1,3))

pixel\_vals = np.float32(pixel\_vals)

criteria = (cv2.TERM\_CRITERIA\_EPS + cv2.TERM\_CRITERIA\_MAX\_ITER, 100, 0.85)

k = 3

retval, labels, centers = cv2.kmeans(pixel\_vals, k, None, criteria, 10, cv2.KMEANS\_RANDOM\_CENTERS)

centers = np.uint8(centers)

segmented\_data = centers[labels.flatten()]

segmented\_image = segmented\_data.reshape((image.shape))

plt.imshow(segmented\_image)

<end>