**Lab-10 Digital Image Processing**

**Morphological Operations**

**Name:** Ahmad Amjad Mughal

**Reg No:** 121672

**Class:** BSCS-6C

**Task 1**

**Morphological Operation Erosion**

#Morphological Operation Erioson is performed on an image

#Necessary libraries

import PIL

import cv2

from PIL import Image

import numpy as np

#Read an original Image and erode it

originalImage = cv2.imread('Signature.png',0)

kernelSize = np.ones((2,2), np.uint8);

erosion = cv2.erode(originalImage, kernelSize, iterations = 6)

#InvertedImage is the updated image after performed erosion

invertedImage = np.invert(erosion)

modifiedImage = Image.fromarray(invertedImage)

modifiedImage.show()

**Screenshot**

****

**Morphological Operation Dilation**

#Morphological Operation Dilation is performed on an image

#Necessary libraries

import PIL

import cv2

from PIL import Image

import numpy as np

#Read an original Image and dilate it

originalImage = cv2.imread('Signature.png',0)

kernelSize = np.ones((2,2), np.uint8);

dilated = cv2.dilate(originalImage, kernelSize, iterations = 6)

#InvertedImage is the updated image after performed dilation

invertedImage = np.invert(dilated)

modifiedImage = Image.fromarray(invertedImage)

modifiedImage.show()

**Screenshot**

****

**Morphological Operation Opening**

#Inlcude Necessary Libraries

import PIL

import cv2

from PIL import Image

import numpy as np

#Get an original Image and define a Kernel of Size 2\*2

originalImage = cv2.imread('Signature.png',0)

kernelSize = np.ones((2,2), np.uint8);

#Apply Operation Opening that erodes then dilates

opening = cv2.morphologyEx(originalImage, cv2.MORPH\_OPEN, kernelSize)

#Resultant image is then inverted

invertedImage = np.invert(opening)

modifiedImage = Image.fromarray(invertedImage)

modifiedImage.show()

**Screenshot**



**Morphological Operation Closing**

#Inlcude Necessary Libraries

import PIL

import cv2

from PIL import Image

import numpy as np

#Get an original Image and define a Kernel of Size 2\*2

originalImage = cv2.imread('Signature.png',0)

kernelSize = np.ones((2,2), np.uint8);

#Apply Operation Closing that dilates then erodes

closing = cv2.morphologyEx(originalImage, cv2.MORPH\_CLOSE, kernelSize)

#Resultant image is then inverted

invertedImage = np.invert(closing)

modifiedImage = Image.fromarray(invertedImage)

modifiedImage.show()

**Screenshot**

****

**Task 2**

**#Necessary Libraries**

**import matplotlib**

**import matplotlib.pyplot as plt**

**import cv2**

**import PIL**

from PIL import Image

import numpy

#Get and original image and define kernel size of 2\*2

originalImage = cv2.imread('Image2.jpg',0)

kernelSize = numpy.ones((2,2),numpy.uint8)

#Set threshold ar grey value 50 all pixels less than 50 is assigned 0 else 1

ret,thresholdImage = cv2.threshold(originalImage,50,255,cv2.THRESH\_BINARY)

#Perfomed operations closing first then dilation then erosion

closing = cv2.morphologyEx(thresholdImage, cv2.MORPH\_CLOSE, kernelSize)

dilation = cv2.dilate(closing,kernelSize,iterations = 1)

erosion = cv2.erode(dilation,kernelSize,iterations = 5)

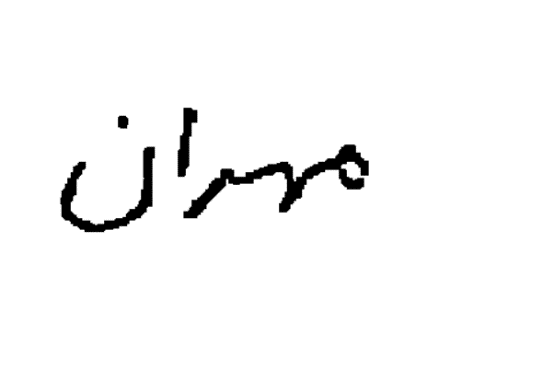
#resultant Image

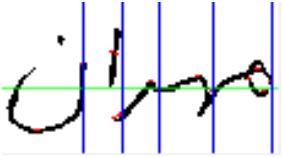
img1 = Image.fromarray(erosion)

img1.show()

**Screenshot**

Original After Operations

****

****

**Why Morphological Operations behave differently at different backgrounds?**

The basic morphological operations like erosion and dilation acts differently on different background images. When an image is in dark background, the dilation behaves differently as compared to white background images.

One of the reasons is Morphological Operations have duality nature. An eroded image with a structuring element whole compliment is equal to the compliment of image dilated with structuring element. This means that erosion operation can be a dilation at some certain backgrounds. Same as for dilation too. Non-Inverting image gets dilated with dilation but an inverted image gets eroded with dilation. This is because of duality nature of morphological operations.