**Lab 09**

|  |  |
| --- | --- |
| Name: | 劉邦彬 |
| Student ID: | R11631033 |
| Total Score: |  |

**Note:**

Most of the explanations in this lab is mandatory, However, giving reasonable explanations to your answer or programs will earn you partial credits when your answer is incorrect.

1. **Multiple Choice (20 points)**

|  |  |  |  |
| --- | --- | --- | --- |
| # | Answer | Explanation (Please write codes to verify it.) | Score |
| 1 | (b) | import numpy as np  import matplotlib.pyplot as plt  import skimage  a = np.array(      [          [1, 1, 1, 1, 1, 1, 1, 1, 1, 1],          [1, 1, 0, 0, 0, 0, 0, 1, 1, 1],          [1, 0, 0, 1, 1, 1, 0, 0, 0, 1],          [1, 0, 0, 1, 1, 1, 1, 0, 0, 1],          [1, 0, 0, 0, 0, 1, 1, 0, 1, 1],          [1, 0, 1, 1, 0, 0, 0, 0, 1, 1],          [1, 0, 1, 1, 1, 1, 1, 0, 0, 1],          [1, 0, 0, 0, 1, 1, 0, 0, 0, 1],          [1, 1, 1, 0, 0, 0, 0, 0, 1, 1],          [1, 1 ,1 ,1, 1, 1, 1, 1, 1, 1,]          ]  )  erosion = skimage.morphology.binary\_erosion(      a, *footprint*=np.ones((2, 2)), *out*=np.zeros\_like(a)  )  plt.figure(*figsize*=(8, 4))  plt.subplot(2, 2, 1)  plt.title("Origin", *fontsize*=16)  plt.imshow(a, *cmap*="gray")  plt.subplot(2, 2, 2)  plt.title("After Erosion", *fontsize*=16)  plt.imshow(erosion, *cmap*="gray")  plt.setp(plt.gcf().get\_axes(), *xticks*=[], *yticks*=[])  plt.tight\_layout()  plt.show()  RESULT:  使用skimage.morphology.binary\_erosion對原A圖進行侵蝕 |  |

1. **Find the Contour (20 points)**

|  |  |  |
| --- | --- | --- |
| # | Description | Score |
| 1 | import numpy as np  import matplotlib.pyplot as plt  import skimage  SingleRice = skimage.io.imread("SingleRice.jpg")  gray\_SingleRice = skimage.color.rgb2gray(SingleRice)  thresh = skimage.filters.threshold\_isodata(gray\_SingleRice)  binary = gray\_SingleRice > thresh  rice\_erosion = skimage.morphology.binary\_erosion(binary, skimage.morphology.diamond(3))  rice\_dilation = skimage.morphology.binary\_dilation(      rice\_erosion, skimage.morphology.diamond(17)  )  rice\_erosion = skimage.morphology.binary\_erosion(      rice\_dilation, skimage.morphology.diamond(23)  )  contours = skimage.measure.find\_contours(rice\_erosion, 0.9)  plt.subplot(1, 2, 1)  plt.imshow(SingleRice)  plt.subplot(1, 2, 2)  plt.imshow(SingleRice)  for contour in contours:      plt.plot(contour[:, 1], contour[:, 0], *color*="red", *linewidth*=2)  plt.setp(plt.gcf().get\_axes(), *xticks*=[], *yticks*=[])  plt.tight\_layout()  plt.show()  RESULT:  首先使用skimage.filters.thershold\_isodat將灰階轉成二值化影像，接著進行二值化影像的侵蝕(erosion)和膨脹(dilation)最後用skimage.measure.find\_contours找到輪廓並用紅色線標記出來 |  |

1. **Image Enhancement (25 points)**

|  |  |  |
| --- | --- | --- |
| # | Description | Score |
| - | Paste your result from (a) to (g) here.  import numpy as np  import matplotlib.pyplot as plt  import skimage  origin = skimage.io.imread("Xray.png", *as\_gray*=True)  plt.figure(*figsize*=(10, 8))  plt.subplot(2, 4, 1)  plt.title("Original image")  plt.imshow(origin, *cmap*="gray")  plt.subplot(2, 4, 2)  a = skimage.filters.laplace(origin)  plt.title("(a)")  plt.imshow(a, *cmap*="gray")  plt.subplot(2, 4, 3)  b = origin - a  plt.title("(b)")  plt.imshow(b, *cmap*="gray")  plt.subplot(2, 4, 4)  *# xray\_sobel\_x = skimage.filters.sobel\_v(b)*  *# xray\_sobel\_y = skimage.filters.sobel\_h(b)*  *# c =b + xray\_sobel\_x + xray\_sobel\_y*  c = skimage.filters.sobel(b)  plt.title("(c)")  plt.imshow(c, *cmap*="gray")  plt.subplot(2, 4, 5)  from scipy.signal import convolve2d  averaging = np.ones((5, 5)) / 25  d = convolve2d(c, averaging, *mode*="same")  plt.title("(d)")  plt.imshow(d, *cmap*="gray")  plt.subplot(2, 4, 6)  e = np.multiply(b, d)  plt.title("(e)")  plt.imshow(e, *cmap*="gray")  plt.subplot(2, 4, 7)  e\_min = np.min(e)  e\_max = np.max(e)  e\_normalized = 255 \* (e - e\_min) / (e\_max - e\_min)  f = origin + e\_normalized  plt.title("(f)")  plt.imshow(f, *cmap*="gray")  plt.subplot(2, 4, 8)  *# f= skimage.exposure.rescale\_intensity(f, out\_range=(0, 255))*  g = skimage.exposure.adjust\_gamma(f, *gamma*=0.5)  plt.title("(g)")  plt.imshow(g, *cmap*="gray")  plt.setp(plt.gcf().get\_axes(), *xticks*=[], *yticks*=[])  plt.tight\_layout()  plt.show()  Result:  首先對原始圖片使用skimage.filters.laplace得(a)  再將原始圖片減去(a)之後得到(b)  再對(b)使用skimage.filters.sobel和原始圖片相加後得到(c)  再使用scipy.signal.convolve2d對(c)和5x5的average filter進行convolved得到(d)  再對(b)和(d)使用np.multply進行element-by-element multiply得到(e)  將原始圖像以及對(e)進行normalized後相加得到(f)  最後對(f)使用skimage.exposure.adjust\_gamma進行power-law transformation得到(g) |  |

1. **Fingerprint Analysis (35 points)**

|  |  |  |
| --- | --- | --- |
| # | Description | Score |
| - | Paste your result here and briefly describe your image processing procedure and approach. How do you think your results are?  import numpy as np  import matplotlib.pyplot as plt  import skimage  from skimage.morphology import erosion ,thin  finger = skimage.io.imread('Fingerprint.tif' ,*as\_gray* = True)  plt.figure(*figsize* = (12,8))  plt.subplot(2,2,1)  plt.title('Original image')  plt.imshow(finger ,*cmap*='gray')  plt.subplot(2,2,2)  thresh = skimage.filters.threshold\_otsu(finger)  binary = finger  > thresh  seed = finger > 70  reconstructed\_image = skimage.morphology.reconstruction(seed, binary   ,*method*='erosion')  reconstructed\_image = skimage.util.invert(reconstructed\_image)  plt.title('Reconstruct image')  plt.imshow(reconstructed\_image ,*cmap*='gray')  plt.subplot(2,2,3)  a = skimage.morphology.skeletonize(reconstructed\_image)  plt.title('Skeletonize image')  plt.imshow(a,*cmap*='gray')  plt.subplot(2,2,4)  plt.title('Features founded')  plt.setp(plt.gcf().get\_axes() ,*xticks*=[] , *yticks*=[])  plt.tight\_layout()  plt.show()    首先二值化之後使用skimage.morphology.reconstruction進行重建再使用skimage.util.invert將重建後影像進行黑白顛倒再使用skimage.morphology.skeletonize將圖片細化。  而最後的Ridge-ending 和Bifurcation我不會 |  |