**Lab 07**

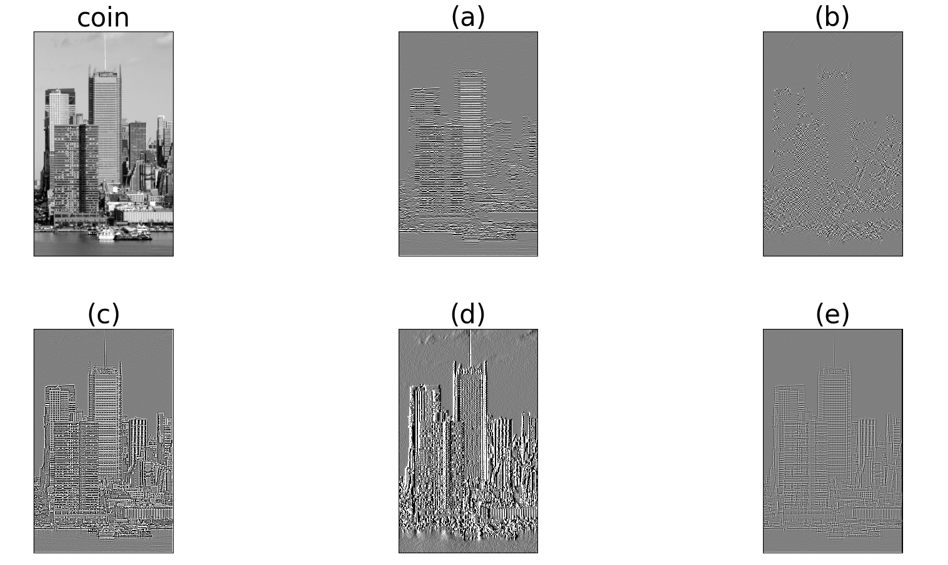
|  |  |
| --- | --- |
| Name: | 劉邦彬 |
| Student ID: | R11631033 |
| Total Score: |  |

**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. **Filters and Convolution (25 points, 5 points each question)**
2. import numpy as np
3. import matplotlib.pyplot as plt
4. import skimage
5. from scipy.signal import convolve2d
6. coin = skimage.io.imread("building.jpg", as\_gray=True)
7. plt.subplot(2, 3, 1)
8. plt.imshow(coin, cmap="gray")
9. plt.title("coin", fontsize=30)
10. plt.subplot(2, 3, 2)
11. a = np.full((3, 3), -1)
12. a[1, :] = 2
13. filter\_a = convolve2d(coin, a, mode="same")
14. plt.imshow(filter\_a, cmap="gray", vmin=-0.1, vmax=0.1)
15. plt.title("(a)", fontsize=30)
16. plt.subplot(2, 3, 3)
17. b = np.full((3, 3), -1)
18. b[(2, 1, 0), (0, 1, 2)] = 2
19. filter\_b = convolve2d(coin, b, mode="same")
20. plt.imshow(filter\_b, cmap="gray", vmin=-0.1, vmax=0.1)
21. plt.title("(b)", fontsize=30)
22. plt.subplot(2, 3, 4)
23. c = np.full((3, 3), -1)
24. c[1, 1] = 8
25. filter\_c = convolve2d(coin, c, mode="same")
26. plt.imshow(filter\_c, cmap="gray", vmin=-0.1, vmax=0.1)
27. plt.title("(c)", fontsize=30)
28. plt.subplot(2, 3, 5)
29. d = np.full((3, 3), 0)
30. d[(0, 2), 0] = -1
31. d[1, 0] = -2
32. d[1, 2] = 2
33. d[(0, 2), 2] = 1
34. filter\_d = convolve2d(coin, d, mode="same")
35. plt.imshow(filter\_d, cmap="gray", vmin=-0.1, vmax=0.1)
36. plt.title("(d)", fontsize=30)
37. plt.subplot(2, 3, 6)
38. e = np.full((3, 3), 0)
39. e[1, (0, 2)] = 1
40. e[(0, 2), 1] = -1
41. filter\_e = convolve2d(coin, e, mode="same")
42. plt.imshow(filter\_e, cmap="gray", vmin=-0.1, vmax=0.1)
43. plt.title("(e)", fontsize=30)
44. plt.setp(plt.gcf().get\_axes(), xticks=[], yticks=[])
45. plt.tight\_layout()
46. plt.show()

**Result:**

****

|  |  |  |
| --- | --- | --- |
| # | Description | Score |
| a | 強調水平類似sobel\_h filter |  |
| b | 強調對角線 |  |
| c | Laplacian 二階微分 |  |
| d | Sobel\_v filter |  |
| e | 類似Laolacian強調邊緣 |  |

1. **Denoising a Picture (15 points)**

|  |  |  |
| --- | --- | --- |
| # | Description | Score |
| - | Be sure to show all your experiment result (e.g., image processing steps and output images) here.  import numpy as np  import matplotlib.pyplot as plt  import skimage  from skimage import img\_as\_float, io  from skimage.morphology import square, disk, square, rectangle, diamond  from skimage.filters.rank import median, minimum, maximum, mean  Saturn = skimage.io.imread("Saturn.jpg")  print(Saturn.shape)  plt.subplot(3, 4, 1)  plt.imshow(Saturn)  plt.title("Saturn", fontsize=30)  plt.subplot(3, 4, 2)  ball = skimage.morphology.ball(radius=5)  mean\_img = skimage.filters.rank.mean(Saturn, ball)  plt.imshow(mean\_img)  plt.title("Mean filter", fontsize=30)  plt.subplot(3, 4, 3)  gaussian\_img = skimage.filters.gaussian(Saturn, sigma=1, channel\_axis=-1)  plt.imshow(gaussian\_img)  plt.title("Gaussian filter", fontsize=30)  plt.subplot(3, 4, 4)  tv\_img = skimage.restoration.denoise\_tv\_chambolle(Saturn, weight=0.2, channel\_axis=-1)  plt.imshow(tv\_img)  plt.title(" Total variation filter", fontsize=30)  plt.subplot(3, 4, 5)  bilateral\_img = skimage.restoration.denoise\_bilateral(      Saturn, sigma\_color=0.1, sigma\_spatial=15, channel\_axis=-1  )  plt.imshow(bilateral\_img)  plt.title(" Bilateral filter", fontsize=30)  plt.subplot(3, 4, 6)  Wavelet\_img = skimage.restoration.denoise\_wavelet(      Saturn, channel\_axis=-1, convert2ycbcr=True, rescale\_sigma=True  )  plt.imshow(Wavelet\_img)  plt.title(" Wavelet denoising filter", fontsize=30)  Saturn = img\_as\_float(io.imread("Saturn.jpg", as\_gray=True))  neighborhood = square(width=3)  Saturn\_smoothed = img\_as\_float(minimum(Saturn, neighborhood))  detail = Saturn - Saturn\_smoothed  coins\_sharpend = Saturn + detail \* 2  plt.subplot(3, 4, 7)  plt.imshow(Saturn\_smoothed, cmap="gray", vmin=0, vmax=1)  plt.title("Smoothed image", fontsize=30)  plt.subplot(3, 4, 8)  plt.imshow(detail, cmap="gray", vmin=-0.5, vmax=1)  plt.title("Detail", fontsize=30)  plt.subplot(3, 4, 9)  plt.imshow(coins\_sharpend, cmap="gray", vmin=0, vmax=1)  plt.title("Sharpened", fontsize=30)  Saturn\_laplacian = skimage.filters.laplace(Saturn)  plt.subplot(3, 4, 10)  plt.title("Laplacian", fontsize=30)  plt.imshow(Saturn\_laplacian, cmap="gray", vmin=-0.1, vmax=0.1)  Saturn\_enhanced = Saturn + Saturn\_laplacian \* 2  plt.subplot(3, 4, 11)  plt.title("Enhanced image", fontsize=30)  plt.imshow(Saturn\_enhanced, cmap="gray", vmax=1.0, vmin=0)  plt.setp(plt.gcf().get\_axes(), xticks=[], yticks=[])  plt.tight\_layout()  plt.show()  Result:  使用skimage.filteru以及skimage.restoration中的各個filter去除雜訊 |  |

1. **Properties of Convolution (20 points)**

|  |  |  |
| --- | --- | --- |
| # | Description | Score |
| 1 |  |  |
| 2 | Median filter是一個非線性濾波器,將mask遮到的數據由小到大排序，對數列取中值後再取代中間數據 |  |

1. **Image Segmentation and Color Space (20 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 diamond  y\_fan = skimage.io.imread("YellowFan.png")  img = np.zeros((y\_fan.shape[0], y\_fan.shape[1]))  y\_fanhsv = skimage.color.rgb2hsv(y\_fan)  img[np.logical\_and(y\_fanhsv[:, :, 0] > 0.15, y\_fanhsv[:, :, 0] < 0.19)] = 255  fan\_erosion = skimage.morphology.binary\_erosion(img, diamond(5))  fan\_dilation = skimage.morphology.binary\_dilation(fan\_erosion, diamond(5))  f, (ax0, ax1) = plt.subplots(1, 2)  ax0.imshow(y\_fan)  ax1.imshow(fan\_dilation, cmap="gray")  plt.tight\_layout()  plt.setp(plt.gcf().get\_axes(), xticks=[], yticks=[])  labeled\_mask = skimage.measure.label(fan\_dilation)  fan\_properties = skimage.measure.regionprops(labeled\_mask)  fan\_area\_size = fan\_properties[0].area  print(f"Area size of the yellow fan: {fan\_area\_size} pixels")  plt.show()  Result:  利用上課範例將圖片轉換成hsv後調整參數找出黃色的區域之後再做erosion和dilation並用skimage.measure.resionprops算出面積為 |  |

1. **Document Scanner (20 points)**

|  |  |  |
| --- | --- | --- |
| # | Description | Score |
| - | Paste your result here and briefly describe your image processing procedure and approach. How do you think your results are?  from skimage import transform  import matplotlib.pylab as plt  import numpy as np  import skimage  invoice = skimage.io.imread("invoice.jpg")  src = np.array([[0, 0], [0, 563], [422, 563], [422, 0]])  dst = np.array([[120, 200], [75, 520], [290, 520], [280, 190]])  tform3 = transform.ProjectiveTransform()  tform3.estimate(src, dst)  warped = transform.warp(invoice, tform3, output\_shape=(563, 422))  gray\_invoice = skimage.color.rgb2gray(warped)  local\_threshold = skimage.filters.threshold\_local(      gray\_invoice, block\_size=95, method="gaussian"  )  otsu\_thresh = skimage.filters.threshold\_otsu(gray\_invoice)  otsu\_scanned = gray\_invoice > otsu\_thresh  yen\_thresh = skimage.filters.threshold\_yen(gray\_invoice)  yen\_scanned = gray\_invoice > otsu\_thresh  plt.subplot(1, 4, 1)  plt.imshow(invoice, cmap="gray")  plt.title("Origin image" ,fontsize =30)  plt.plot(dst[:, 0], dst[:, 1], ".r")  plt.subplot(1, 4, 2)  plt.imshow(warped, cmap=plt.cm.gray)  plt.title("Intermediate image" ,fontsize =30)  plt.subplot(1, 4, 3)  plt.imshow(otsu\_scanned, cmap="gray" ,vmin=0 ,vmax=1)  plt.title("Scanned invoice\_otsu",fontsize =30)  plt.subplot(1, 4, 4)  plt.imshow(yen\_scanned, cmap="gray",vmin=0 ,vmax=1)  plt.title("Scanned invoice\_yen",fontsize =30)  plt.setp(plt.gcf().get\_axes(), xticks=[], yticks=[])  plt.tight\_layout()  plt.show()  Result:  首先使用skimage.transform.wrap 做找發票四個角點(紅點處)然後做透視變換，再透過skimage.filters.try\_all\_thershold設定thersohold讓影像有黑白的感覺挑出感覺不錯的ostu和yen |  |