Hw2 Image Sharpening

資工四 陳昱瑋 409410118

Date due: 2024/4/29

Date handed in: 2024/5/4

Technical description

- **Foundation of Sharpening Filters**

  Sharpening filters 可以透過一次微分(First Derivative)與二次微分(Second Derivative)實現。也就是透過 spatial differentiation，增加邊緣的銳利度，減少低 gray-level 變化的權重。

  (1) First Derivative

  Nonzero 會出現在梯度變化為非 0 的 ramp，因此對於 gray-level 變化會有較敏感的反應。也就是在 gray-level 變化快速時，對 gray-level 做 first derivative 之曲線會有大幅度的變化。

  (2) Second Derivative

  若 gray-level 在某一區之變化與另一區之變化很大，則為 Nonzero 的 ramp。有就是對於灰階變化大的區域：畫面的細節部分，回有較敏感的反應。原本已經很清楚的細節不會再做其他工作，針對黑暗中或是高光中的細節進行加強。

- **Laplacian Operator**

  進行 Second Derivative，進行細節的加強：

$$\nabla^2 f = \frac{\sigma^2 f}{\sigma^2 x} + \frac{\sigma^2 f}{\sigma^2 y}$$

$$\frac{\sigma^2 f}{\sigma^2 x} = f(x+1,y) + f(x-1,y) - 2f(x,y)$$

$$\frac{\sigma^2 f}{\sigma^2 y} = f(x,y+1) + f(x,y-1) - 2f(x,y)$$

$$\nabla^2 f = f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)$$

  ⇨ Filter mask: $\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$

- Image enhancement by Laplacian Operator

$$g(x) = f(x,y) - \nabla^2 f(x,y)$$

$$= \begin{cases} 5f(x,y) - [f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1)], \\ \quad if\ the\ center\ coefficent\ of\ the\ Laplacian\ mask\ is\ negative. \\ 5f(x,y) + [f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1)], \\ \quad if\ the\ center\ coefficent\ of\ the\ Laplacian\ mask\ is\ positive. \end{cases}$$

- **Unsharp Masking and High-Boost Filtering**

  假設 $\bar{f}(x,y)$ 為 input image 經過模糊處理後的影像，則：

$$f_s(x,y) = f(x,y) - \bar{f}(x,y)$$

  為 sharpened image。

- Image enhancement by High-boost filtering

$$\begin{aligned} f_{hb}(x,y) &= Af(x,y) - \bar{f}(x,y) \\ &= (A-1)f(x,y) + f(x,y) - \bar{f}(x,y) \\ &= (A-1)f(x,y) + f_s(x,y) \end{aligned}$$

$f_s(x,y)$可以透過 second derivative 達成：

$\Rightarrow f_{hb}(x,y)$

$$= \begin{cases} (A-1)f(x,y) - \nabla^2 f(x,y), \\ if\ the\ center\ coefficent\ of\ the\ Laplacian\ mask\ is\ negative. \\ (A-1)f(x,y) + \nabla^2 f(x,y), \\ if\ the\ center\ coefficent\ of\ the\ Laplacian\ mask\ is\ positive. \end{cases}$$

Execution process

- **functions**

1) Laplacian(img)
   - ⇨ 進行 Laplacian operator
   - ⇨ img: input image
2) Laplacian_with_hb(img)
   - ⇨ 進行 High-Boost，並透過 Second derivative 取得模糊化的 image
   - ⇨ img: input image
3) plot(img, masked, filtered, title)
   - ⇨ 呈現 original image, masked image, filtered image
   - ⇨ image: original image
   - ⇨ masked: masked image
   - ⇨ filtered: filtered image

- **Laplacian operator process**

  Import image to Laplacian operator => Do Laplacian operator(with the filter $\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$) to get masked image => Do image enhancement by the masked image

- **High-Boost process**

  Import image to High-Boost process => Suppose the blurred image can obtain by the second derivative, get the sharpened image with Laplacian operator => Multiply constant “A” to the original image numpy array => Do image enhancement $Af(x) - f_{blurred}(x)$

- **Execution**
  - Laplacian operator

    ```
    python Laplacian.py
    ```

    (You can modify the mask_type as you want)
  - High-Boost process

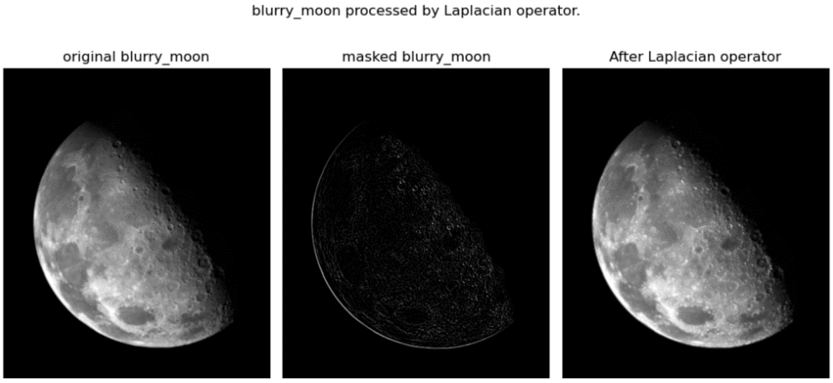
    ```
    python High-Booost.py
    ```

    (You can modify the A as you want)

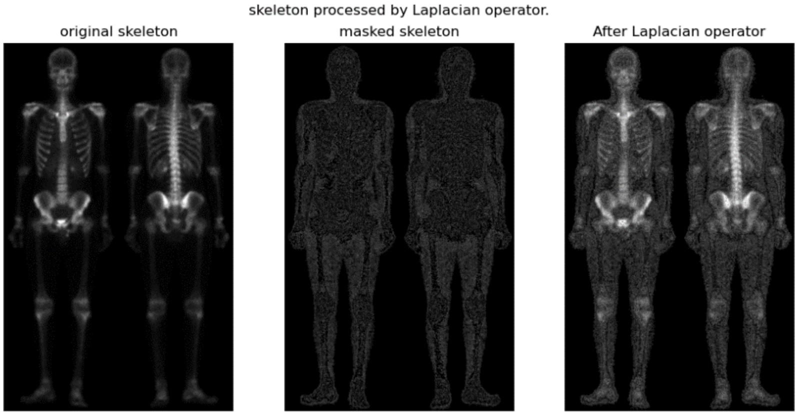
Experimental result

1. Laplacian Operator

- blurry moon processed by Laplacian operator.

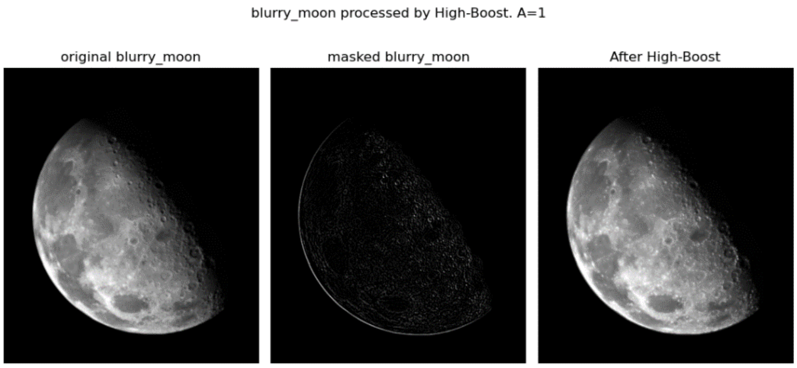


- skeleton processed by Laplacian operator.

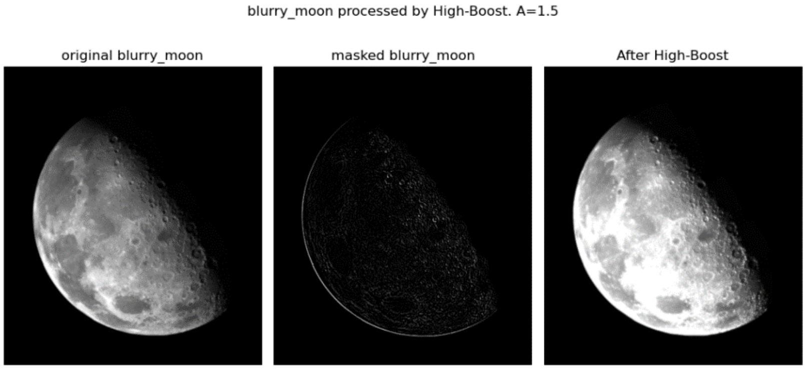


2. Unsharp Masking and High-Boost Filter

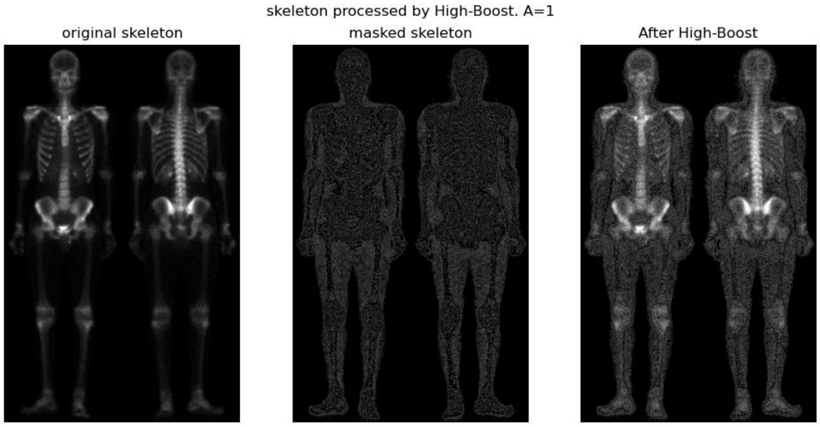
- blurry moon processed by High-Boost with A=1.



- blurry moon processed by High-Boost with A=1.5.

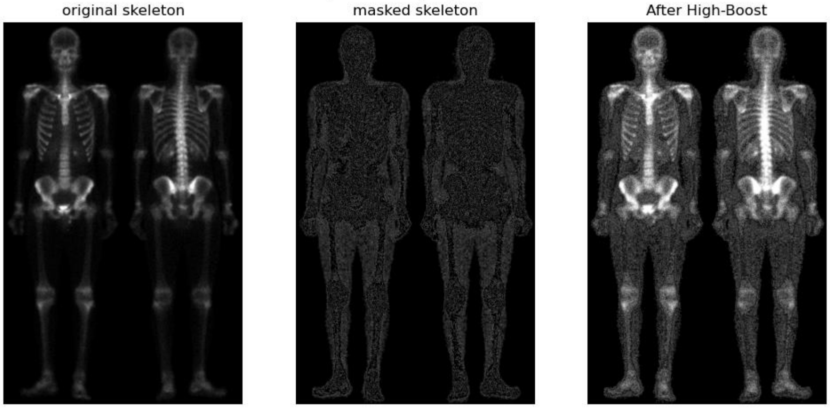


- skeleton processed by High-Boost with A=1.



- skeleton processed by High-Boost with A=1.5.

skeleton processed by High-Boost. A=1.5



original skeleton

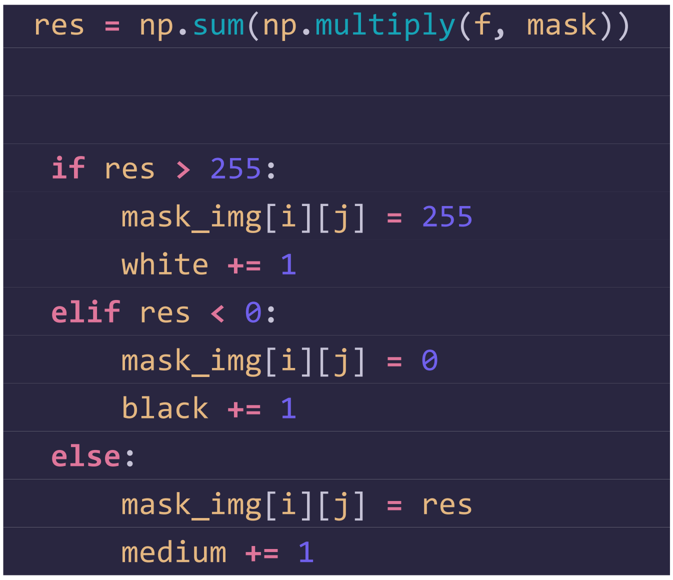
masked skeleton

After High-Boost

Discussions

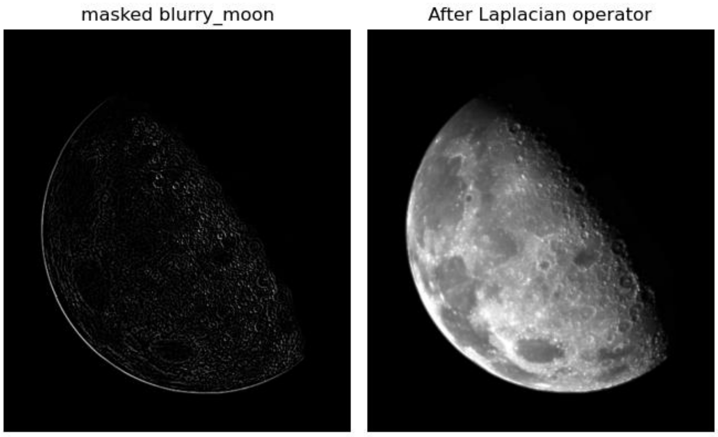
- ***The impact of the different filtered masks on same image***

  針對每張圖有做簡單的分析，對$\nabla^2 f(x,y)$進行數值上的分類：



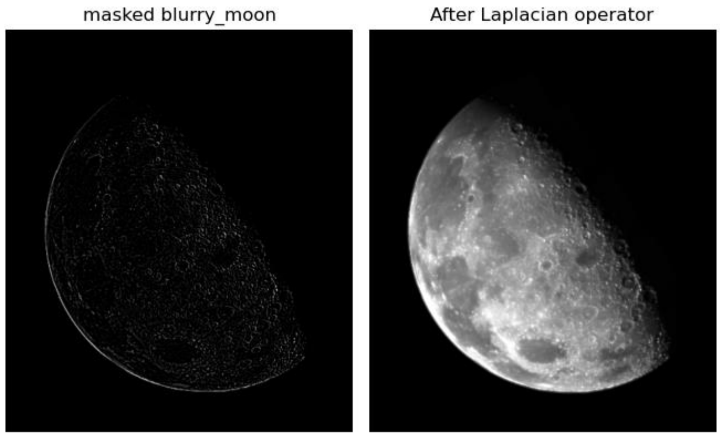
```
res = np.sum(np.multiply(f, mask))


  if res > 255:
      mask_img[i][j] = 255
      white += 1
  elif res < 0:
      mask_img[i][j] = 0
      black += 1
  else:
      mask_img[i][j] = res
      medium += 1
```

  - ○ filtered mask = $\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$



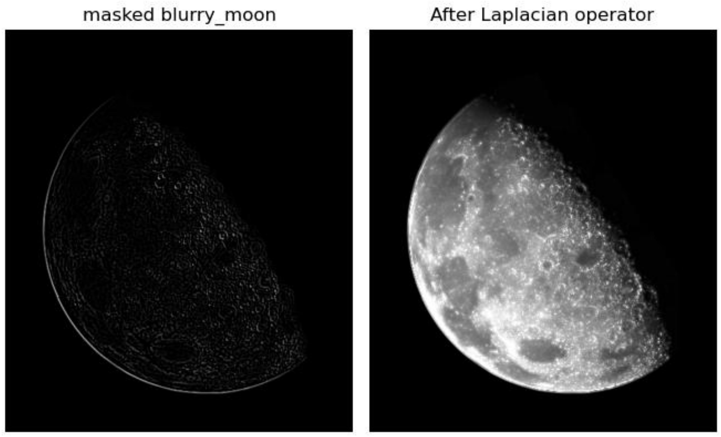
```
white:  0
medium:  207138
black:  42494
```

  - ○ filtered mask = $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$



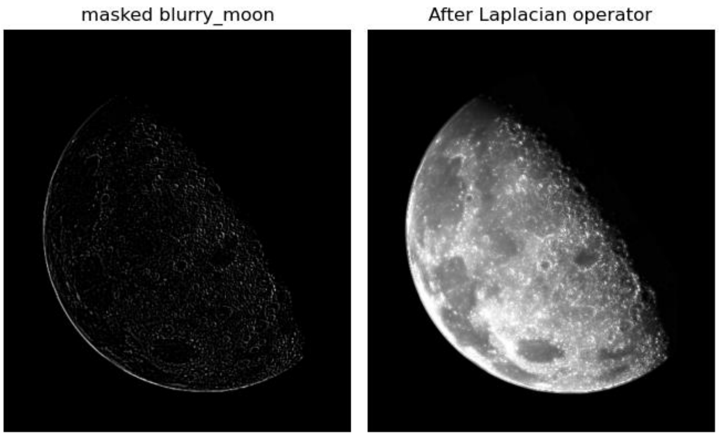
```
white:  3
medium:  200207
black:  49425
```

- filtered mask = $\begin{bmatrix}1 & 1 & 1\\ 1 & -8 & 1\\ 1 & 1 & 1\end{bmatrix}$



```
white:  0
medium:  204913
black:  44729
```

- filtered mask = $\begin{bmatrix}-1 & -1 & -1\\ -1 & 8 & -1\\ -1 & -1 & -1\end{bmatrix}$



```
white:  0
medium:  197479
black:  52150
```

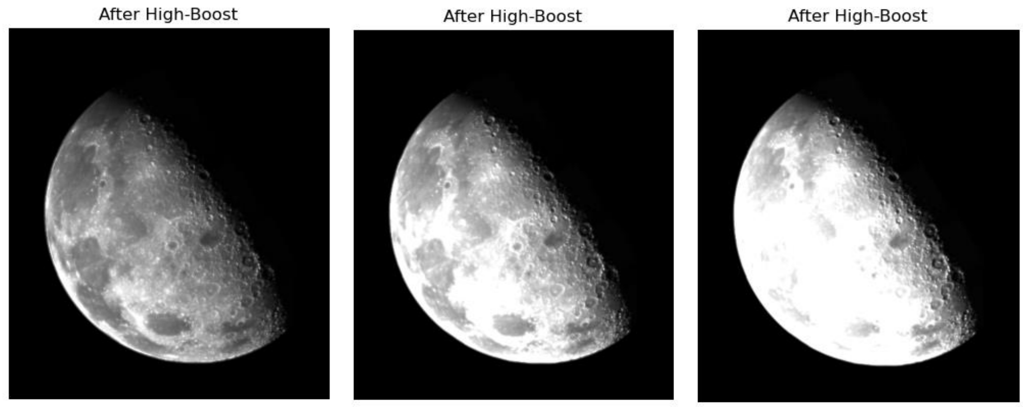
雖然光看圖片並不能完全分辨之間的差別，但可以明顯看出來 the center coefficient of Laplacian filter(k) +/-8 與+/-4 的差別，+/-8 最後的 enhancement image 有較多的細節。

從簡單的數值 cluster 可以看出在+/-8 的 masked image，gray-level 在 medium level 有較多的數量，而+/-4 的 masked image，gray-level 在 black 有較多的數量。可以推測，k=+/-8 針對細節做更多的加強，而 k=+/-4 針對邊緣(edge)會做更多加強。

在正負值的差別，以 k=8 與 k=-8 為例，k=8 的 medium/black 差值(197479 - 52150)為 145,329，k=-8 的差值(204913 - 44729)為 160,184，k=-8 具較多集中在 medium level，因此推測負值對細節較敏感，且其差值較大，且觀察下表，|k|越大其正負值之 medium-black 會有更明顯的差值，有更強烈的 enhancement effect。

k	medium	black	medium-black
4	200207	49425	150,782
-4	207138	42494	164,644
8	197479	52150	145,329
-8	204913	44719	160,194

- ***The impact of the constant “A” in the High-Boost process***



The enhancement image with constant A=1, 1.5, 2, respectively.

由上圖可以發現，當 constant A 越大，$f_{hb}(x,y) = Af(x,y) - \bar{f}(x,y)$，原圖的 gray-level 分布會更趨近於 255，灰階與亮部會變得更白，因此可以很明顯地看到 enhancement image 隨著 A 變大，整體高光所佔比例會越來越大。

References and Appendix

Matplotlib documentation — Matplotlib 3.8.4 documentation

Matplotlib Subplot (w3schools.com)

NumPy documentation — NumPy v1.26 Manual

https://ecourse2.ccu.edu.tw/mod/resource/view.php?id=857335