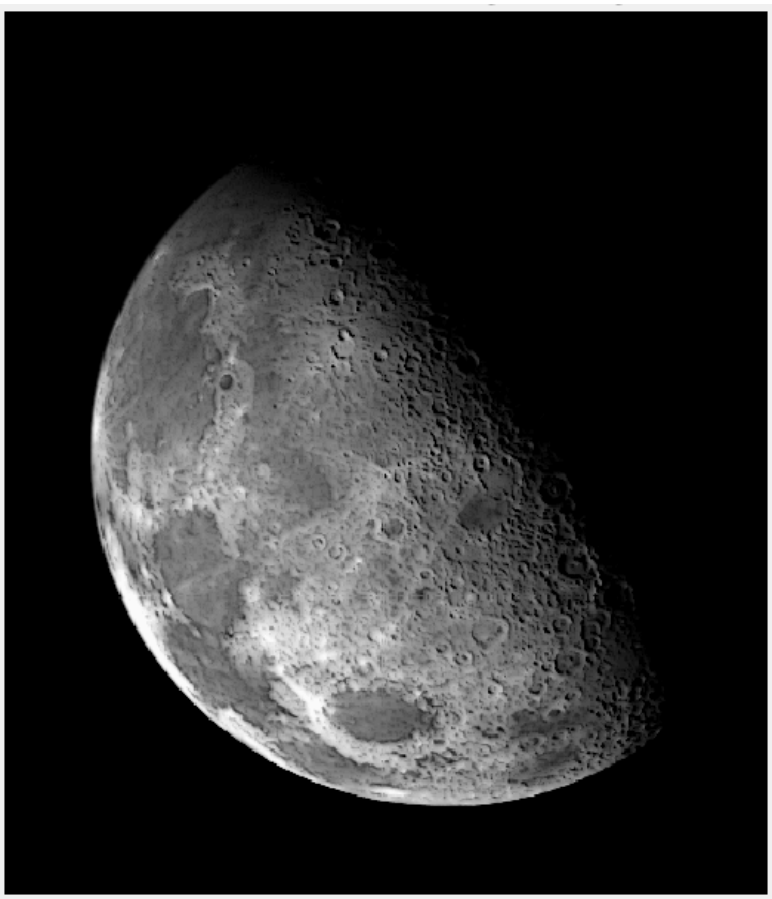
Image Processing Homework #2

Implementations of Laplacian Operator, Unsharp Masking, and High-Boost Filtering in Spatial Domain and Frequency Domain



Advisor : Jin-Jang Leou (柳金章)
Student : Chih-Chia Lo (羅治嘉)
Registration Number : 610410002
Date Due : 2021/12/06 23:59
Date Handed In : 2021/11/30 01:30

Contents

I. Technical Description

A. Laplacian Operator in Spatial Domain

The response of a 2-D isotropic filter (or a rotation-invariant filter) is independent of the direction of the discontinuities in the image to which the filter is applied. [1] To process Laplacian Operator, we implement 3 x 3 convolutions on the input images. Laplacian Operator and its extended mask are as follows in Fig. 1.:

(a)

0	1	0
1	-4	1
0	1	0

(b)

1	1	1
1	-8	1
1	1	1

Fig. 1. (a) Laplacian Operator. (b) Extension of Laplacian Operator.

The figure below shows the source code of Laplacian Operator. We take input images and two masks into handcrafted function respectively and do convolutions so that we can compare the results.

```
 9    mask_1 = [0 1 0; 1 -4 1; 0 1 0];
10    mask_2 = [1 1 1; 1 -8 1; 1 1 1];
13    Laplacian_img = uint8(myLaplacian(moon_img, mask_1));
34    Laplacian_img = uint8(myLaplacian(double(moon_img), mask_2));
53    Laplacian_img = uint8(myLaplacian(skeleton_img, mask_1));
74    Laplacian_img = uint8(myLaplacian(double(skeleton_img), mask_2));
216   function [output_img] = myLaplacian(input_img, mask)
217       output_img = zeros(size(input_img));
218       [row,col] = size(input_img);
219       for i = 1 : row - 2
220           for j = 1 : col - 2
221               output_img(i,j) = sum(sum(mask.*input_img(i:i+2,j:j+2)));
222           end
223       end
224   end
```

Fig. 2. The source code of Laplacian Operator and handcrafted convolution.

B. Unsharp Masking and High-Boost Filtering

Unsharp masking consists of generating a sharp image by subtracting from an image a blurred version of itself, i.e.,

$$f_s(x, y) = f(x, y) - \bar{f}(x, y) \quad (1)$$

where $f_s(x, y)$ denotes the unsharp masking image and $\bar{f}(x, y)$ denotes the blurred version of $f(x, y)$. Use Laplacian Operator in Fig. 1. as a low-pass filter. Unsharp masking result can be obtained by subtracting low-pass filter from the ground truth. The addition or the subtraction between the ground truth and the low-pass filter depends on the center coefficient of Laplacian Operator:

$$g(x,y)=\begin{cases} f(x,y)-\nabla^2 f(x,y) & \text{if the center coefficient of the Laplacian mask is negative} \\ f(x,y)+\nabla^2 f(x,y) & \text{if the center coefficient of the Laplacian mask is positive.} \end{cases} \quad (2)$$

where $g(x, y)$ denotes the sharpened image and $\nabla^2 f(x, y)$ denotes the blurred version of $f(x, y)$. If constant A is applied, high-boost filtered image, f_{hb}, is defined as:

$$f_{hb}(x, y) = Af(x, y) - \bar{f}(x, y) \quad (3)$$

where $A \geq 1$ and $\bar{f}(x, y)$ denotes the blurred version of $f(x, y)$. From equation (2), we can infer the formula of high-boost filtering:

$$f_{hb} = \begin{cases} Af(x,y) - \nabla^2 f(x,y), & \text{if the center coefficient of the Laplacian mask is negative,} \\ Af(x,y) + \nabla^2 f(x,y), & \text{if the center coefficient of the Laplacian mask is positive.} \end{cases} \quad (4)$$

The following figure demonstrates the source code of unsharp masking (A = 1) and high-boost filtering (A ≥ 1):

```
8     A = 1.7;
15    unsharp = moon_img.*1-Laplacian_img;
16    high_boost = moon_img.*A-Laplacian_img;
55    unsharp = skeleton_img.*1-Laplacian_img;
56    high_boost = skeleton_img.*A-Laplacian_img;
```

Fig. 3. The source code of Laplacian Operator and handcrafted convolution.

C. Image Enhancement in Frequency Domain

First of all, it is necessary to do padding, otherwise, wraparound error will occur. Secondly, 2-D DFT is used to process input image and Laplacian mask in frequency domain, it is defined as follow:

$$F(u,v) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi(ux/M+vy/N)} \tag{5}$$

where $u = 0, 1, 2, \ldots, M-1$ and also $v = 0, 1, 2, \ldots, N-1$. Then, multiply the results of input image and Laplacian mask, this step is as same as doing convolution in spatial domain. The formula is displayed in equation (2):

$$f(x,y) * h(x,y) \Leftrightarrow F(u,v)H(u,v) \tag{6}$$

where $f(x, y)$ and $h(x, y)$ denote input image and Laplacian mask in spatial domain, respectively, and $F(x, y)$ and $H(x, y)$ denote input image and Laplacian mask in frequency domain. Finally, we use ifft2 function to do inverse 2-D DFT to acquire result:

$$f(x,y) = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)e^{j2\pi(ux/M+vy/N)} \tag{7}$$

Here are the source code of the steps above:

```
93    %Frequency Domain
94    PQ = paddedsize(size(moon_img));
95    mask_f1 = fft2(double(mask_1), PQ(1), PQ(2));
96    image_f1 = fft2(double(moon_img), PQ(1), PQ(2));
97    fft_result = mask_f1 .* image_f1;
98    fft_result = ifft2(fft_result);
99    fft_result = uint8(fft_result(1:size(moon_img,1),1:size(moon_img,2)));
100   unsharp = moon_img.*1-fft_result;
101   high_boost = moon_img.*A-fft_result;
226   function PQ = paddedsize(AB, CD, ~)
227       if nargin == 1
228           PQ = 2 * AB;
229       elseif nargin == 2 && ~ischar(CD)
230           PQ = AB + CD - 1;
231           PQ = 2 * ceil(PQ / 2);
232       elseif nargin == 2
233           m = max(AB); % Maximum dimension.
234           P = 2^nextpow2(2*m); % Find power-of-2 at least twice m.
235           PQ = [P, P];
236       elseif nargin == 3
237           m = max([AB CD]); %Maximum dimension.
238           P = 2^nextpow2(2 * m);
239           PQ = [P, P];
240       else
241           error('Wrong number of inputs.')
242       end
243   end
```

Fig. 4. The source code of Laplacian Operator and handcrafted convolution.

Knowing that Unsharp Masking and High-Boost Filtering are the same afore-mentioned in frequency domain in part B.

D. Other Programming Processes

First of all, take input images that are in HW2_test_images folder and assign these two input images to each variable (Fig. 5.). Secondly, take input images into the handcrafted Laplacian function (Fig. 2.). After that, we process unsharp masking and high-boost filtering in Fig. 3.. Finally, plot the results of Laplacian Operator, Unsharp Masking, and High-Boost Filtering respectively in Fig. 5. and Fig. 6..

```
5    moon_img = double(imread('HW2_test_image/blurry_moon.tif'));
6    skeleton_img = double(imread('HW2_test_image/skeleton_orig.bmp'));
```

Fig. 5. Read Input images and assign variables.

```
18   f = figure('Name','Moon','NumberTitle','off');
19   f.WindowState = 'maximized';
20   subplot(241);
21   imshow(moon_img);
22   title('Original Image', 'FontSize', 16);
23   subplot(242);
24   imshow(Laplacian_img);
25   title('First Laplacian Operator Image', 'FontSize', 16);
26   subplot(243);
27   imshow(unsharp);
28   title('Unsharp Masking Image', 'FontSize', 16);
29   subplot(244);
30   imshow(high_boost);
31   title('High-boost Sharpened Image', 'FontSize', 16);
39   subplot(245);
40   imshow(moon_img);
41   title('Original Image', 'FontSize', 16);
42   subplot(246);
43   imshow(Laplacian_img);
44   title('Second Laplacian Operator Image', 'FontSize', 16);
45   subplot(247);
46   imshow(unsharp);
47   title('Unsharp Masking Image', 'FontSize', 16);
48   subplot(248);
49   imshow(high_boost);
50   title('High-boost Sharpened Image', 'FontSize', 16);
```

```
58   f = figure('Name','Skeleton','NumberTitle','off');
59   f.WindowState = 'maximized';
60   subplot(241);
61   imshow(skeleton_img);
62   title('Original Image', 'FontSize', 16);
63   subplot(242);
64   imshow(Laplacian_img);
65   title('First Laplacian Operator Image', 'FontSize', 16);
66   subplot(243);
67   imshow(unsharp);
68   title('Unsharp Masking Image', 'FontSize', 16);
69   subplot(244);
70   imshow(high_boost);
71   title('High-boost Sharpened Image', 'FontSize', 16);
79   subplot(245);
80   imshow(skeleton_img);
81   title('Original Image', 'FontSize', 16);
82   subplot(246);
83   imshow(Laplacian_img);
84   title('Second Laplacian Operator Image', 'FontSize', 16);
85   subplot(247);
86   imshow(unsharp);
87   title('Unsharp Masking Image', 'FontSize', 16);
88   subplot(248);
89   imshow(high_boost);
90   title('High-boost Sharpened Image', 'FontSize', 16);
```

Fig. 6. Build subplots for moon.tif and skeleton_orig.bmp.

II. Experimental Results

A. Datasets and Environments

This project contains a dataset of HW2_test_image from ecourse2, which there are two grayscale images given in this dataset. The programming language of this project is Matlab_R2021b.

B. Laplacian Operator, Unsharp Masking and High-Boost Filtering in Spatial Domain

The following figures are Laplacian Operator, Unsharp Masking and High-Boost Filtering moon and skeleton results using different masks (A is set to 1.7):

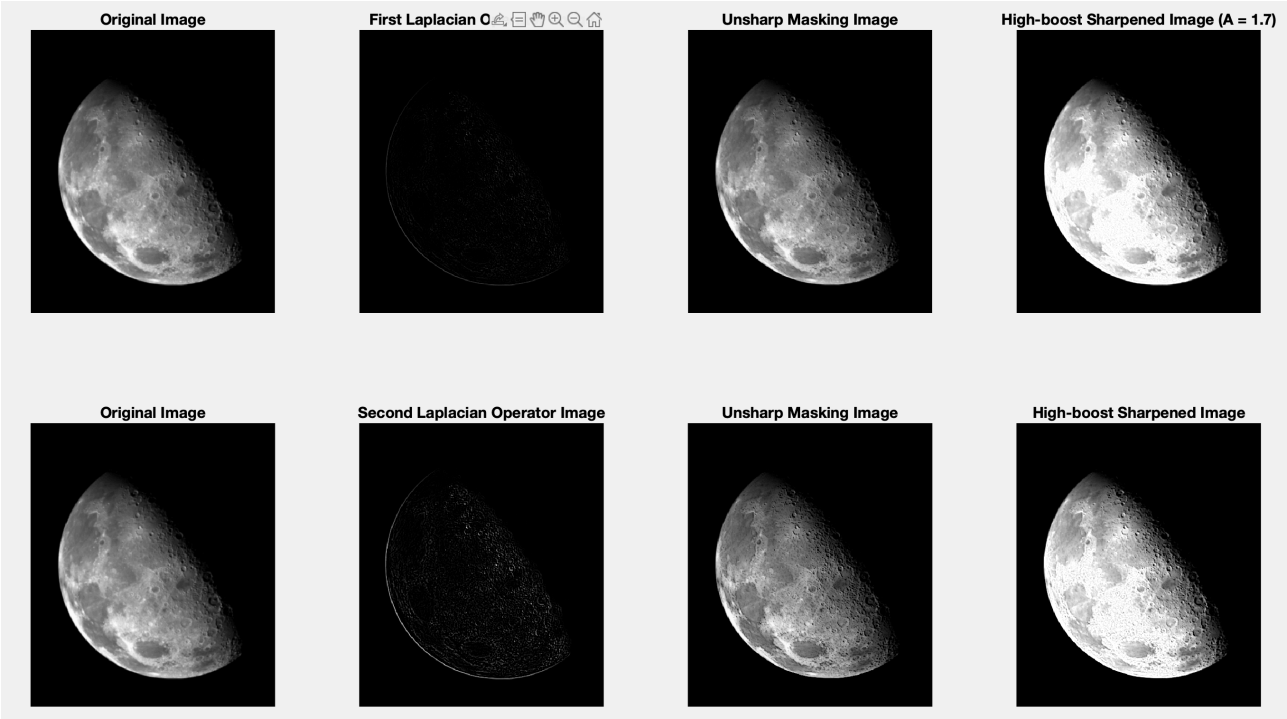


Fig. 7. Results for moon.tif in spatial domain.

where the first column displays the original images, the second column displays the images using different derivative of Laplacian Operator, the third column displays the images using unsharp masking, and the last column displays the images using high-boost filtering.

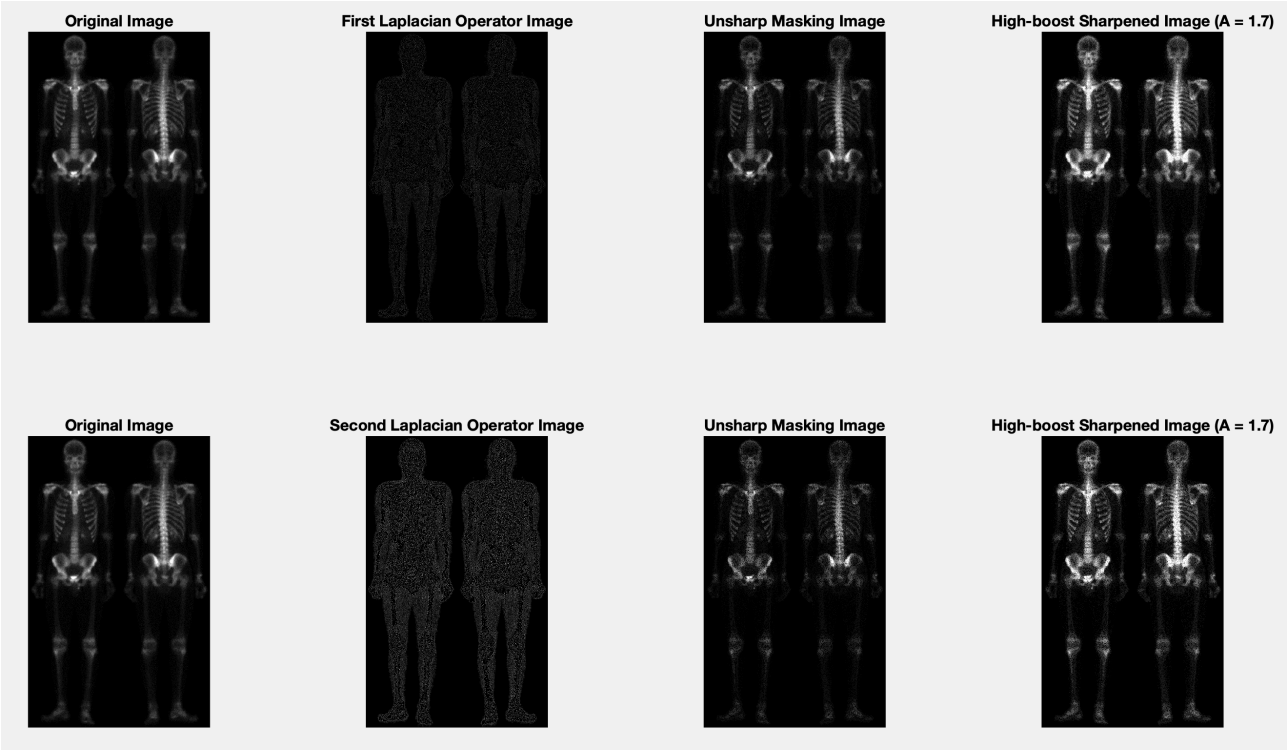


Fig. 8. Results for skeleton_orig.bmp in spatial domain.

where the first column displays the original images, the second column displays the images using different Laplacian Operator masks, the third column displays the images using unsharp masking, and the last column displays the images using High-Boost filtering. The following figure is the comparison using different constant A for High-Boost Filtering:

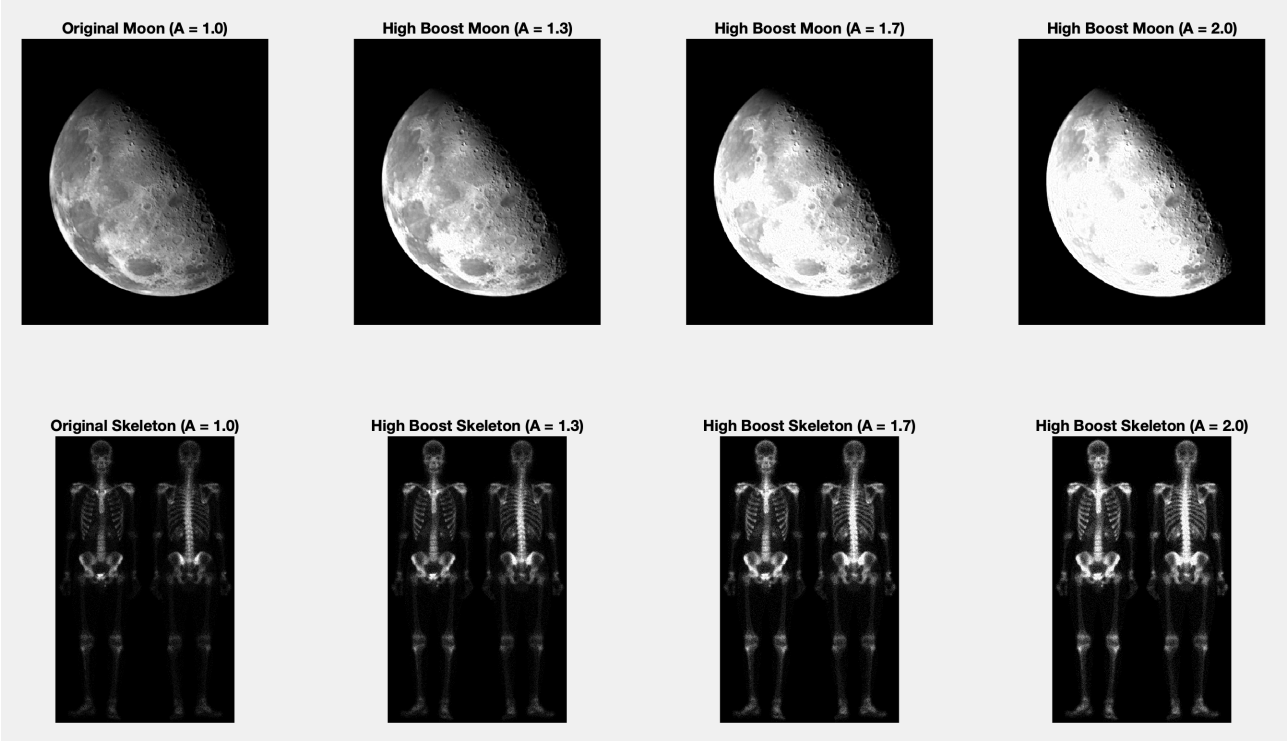


Fig. 9. High–Boost Filtering Comparison.

C. Image Enhancement in frequency domain

The following figures are Laplacian Operator, Unsharp Masking and High-Boost Filtering moon and skeleton results using different masks (*A* is set to 1.7):

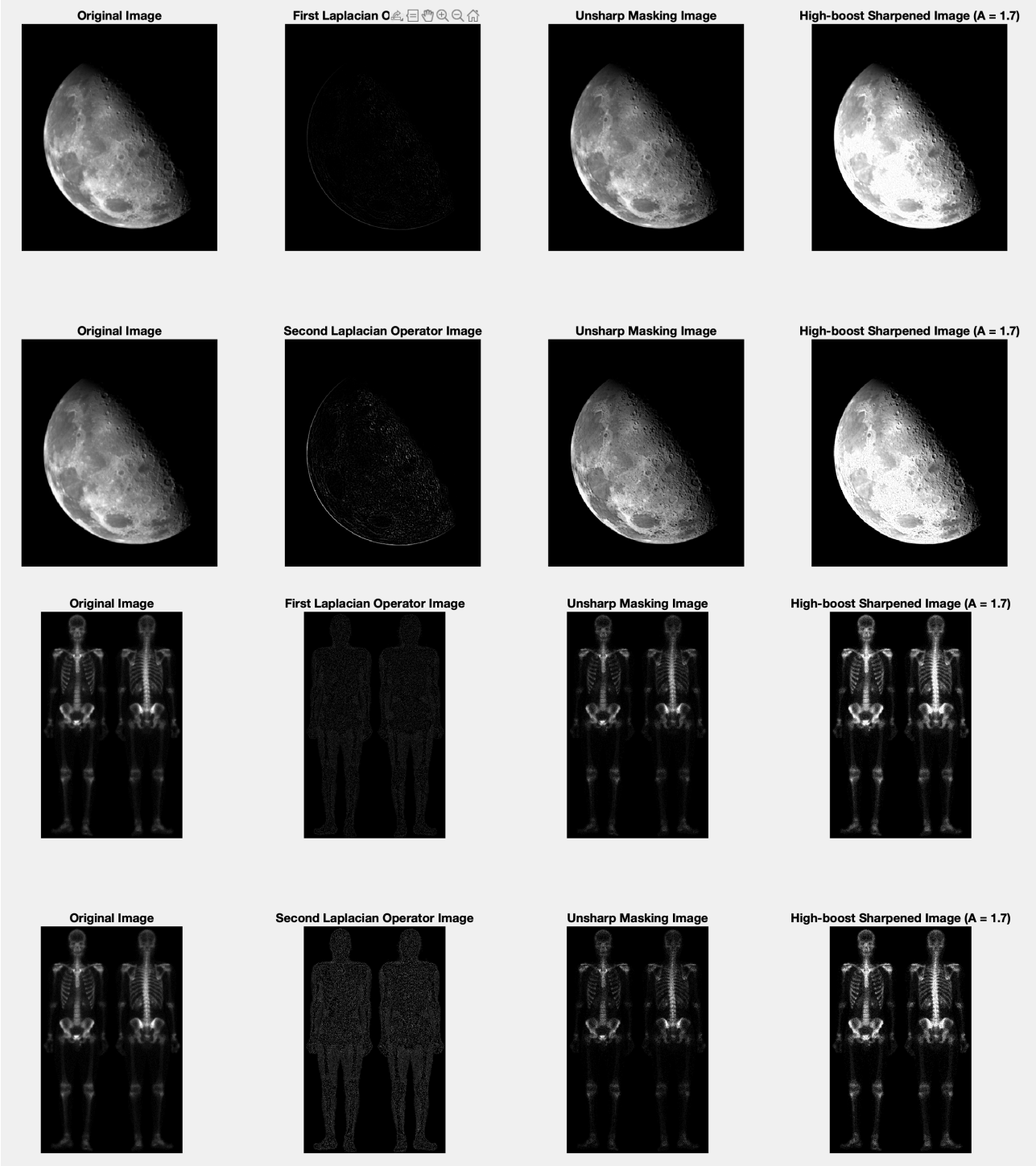


Fig. 10. Results for moon.tif and skeleton_orig.bmp in frequency domain.

III. Discussions

A. Laplacian Operator

Using different Laplacian Operators clearly shows the difference in second column in Fig. 7. and Fig. 8.. The extension of Laplacian mask shows a little bit edges and gray levels, while the results of the original Laplacian mask is hard to see.

B. Unsharp Masking and High-Boost Filtering

Compare with the original Laplacian and the extension of Laplacian, the extension of Unsharp Masking is slightly more clear than the original one. For example, we can see the pits in moon.tif and the ribs in skeleton_orig.bmp have better performances using the extension of Laplacian Operator. Both Laplacian Operators are able to perform well on image enhancement regardless. The equation (1) and (3) are formulae for Unsharp Masking and High-Boost Filtering, when constant A is set to 1, Unsharp Masking is performed, when constant A is larger than 1, then High-Boost Filtering is performed. As constant A increases, the result images will become brighter. All in all, I think both approaches can successfully enhance images.

C. Image Enhancement in frequency domain

As the result of Fig. 10., it is as same as Fig. 7. and Fig. 8.. Therefore, we know that the multiplication in frequency domain has the same effect as the convolution in spatial domain. All in all, this implementation proves equation (6).

IV. References and Appendix

[1] R. C. Gonzalez and R. E. Woods, “Digital Image Processing, 4th Ed., Pearson, New York, NY, 2018.” in Chapter 3 Image Enhancement Gamma Correction.