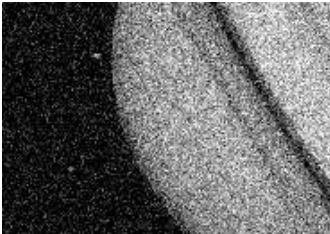




lab11_HW

- lab11_HW
 - Q1
 - Q2

Q1

Write a (not necessary but encouraged GUI) program to denoise the image `11Saturn.jpg`. The program should reduce the noise level of the image by using a Gaussian filter. The Gaussian filter is a very popular filter for denoising. Check `fspecial('gaussian')` for more details. Apply the filter to the image for several times and examine if this helps improving the image quality. You may consider to sharpen the image by applying the Laplacian filter to the denoised image.

(a) Noisy image	(b) Denoised image	(c) Sharpened image
		

just do it as question described

denoised

using `imfilter` and `fspecial('gaussian')`

do ten times to get best solution

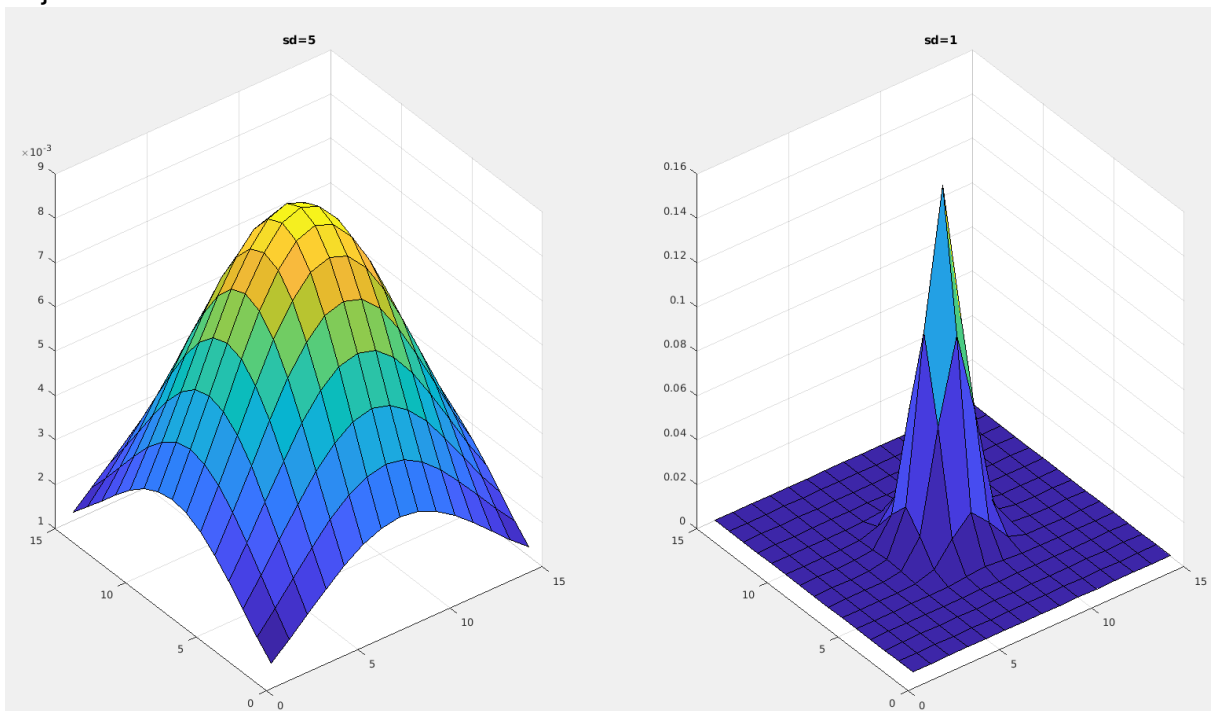
```
denoised_img = img;  
for i = 1:10  
    denoised_img = imfilter(denoised_img, fspecial('gaussian'));  
end
```

dig into gaussian filter

https://en.wikipedia.org/wiki/Gaussian_blur (https://en.wikipedia.org/wiki/Gaussian_blur)

that try different parameter

adjust standard deviation



and you will see that the bigger SD, the more neighborhood will consider in our calculation

and find that sum of gaussian will equal to 1 (no matter the size is 3x3 or 13x13)

the effect 



after knowing that

we can just do it one line

```
denoised_img = imfilter(img, fspecial('gaussian', 10, 10), 'replicate');
```

and also using replicate edge

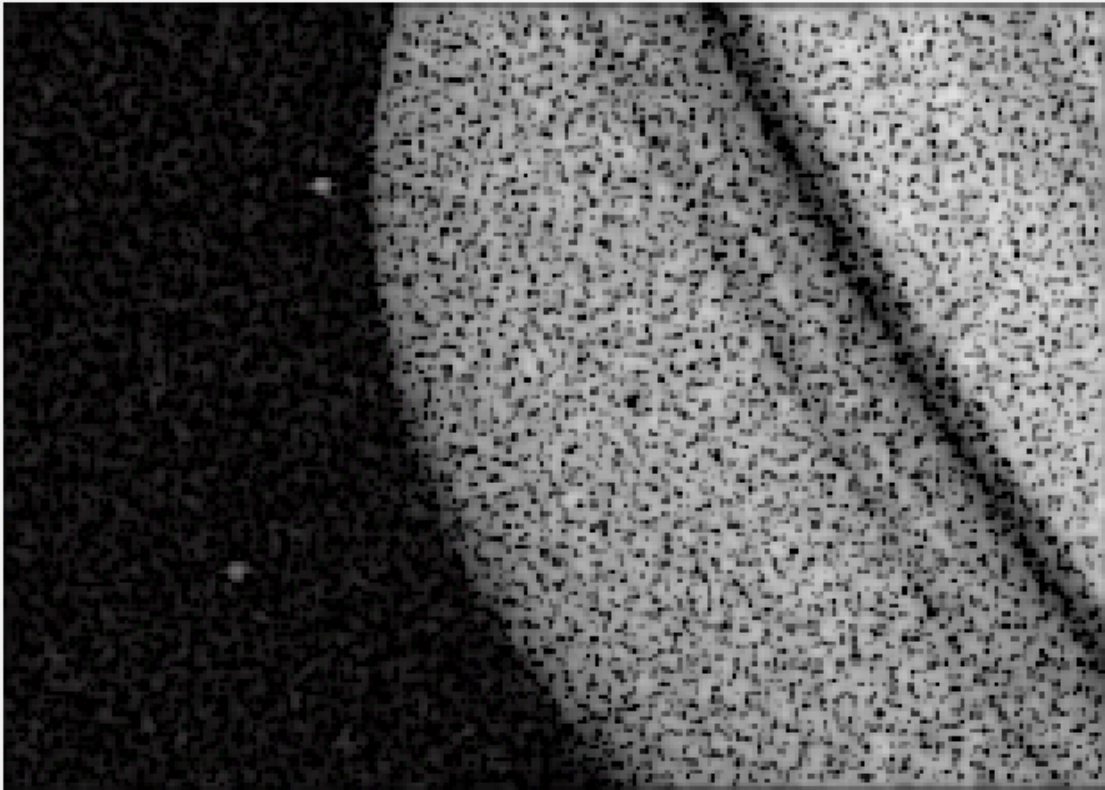
sharpened

use `imfilter fspecial('laplacian')` to get edge

and subtract it to sharpen image

```
sharpened_img = denoised_img - imfilter(denoised_img, fspecial('laplacian', 5, 5), 'replicate');
```

be care for that sharpen cannot do more than two times, it will oversharpen(maybe), noised will show up

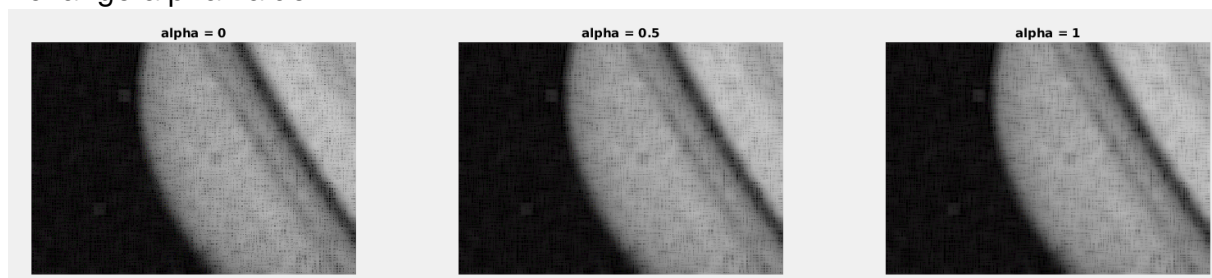


related to laplacian filter

$$\text{Lap}(\alpha) := \frac{1 - \alpha}{1 + \alpha} \cdot \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix} + \frac{\alpha}{1 + \alpha} \cdot \begin{pmatrix} 1 & 0 & 1 \\ 0 & -4 & 0 \\ 1 & 0 & 1 \end{pmatrix}$$

We just can see that alpha 1 is for diagonal and 0 for 4-connect neighbor

I change alpha value

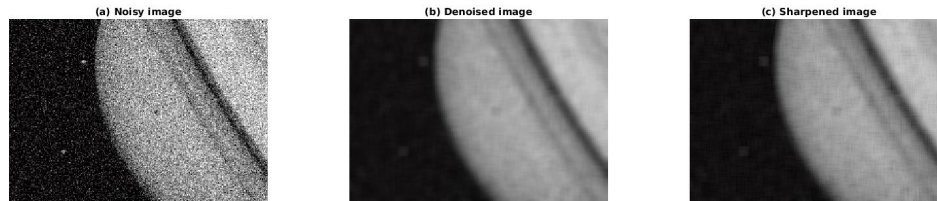


but i cannot tell the difference

the mostly used methods is Laplacian of Gaussian(LoG)
however, this is no related the alpha value

summery

```
see noise_filter.m  
see filter_noise.jpg  
>> noise_filter();
```



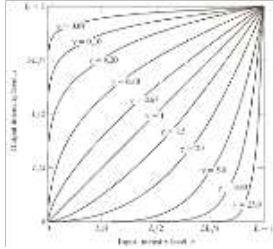
Q2

Successful image enhancement is typically not achieved using a single operation. Rather a range of techniques are combined in order to achieve a final result. This example will focus on enhancing an image of bone scan. Write a (GUI) program to analyze the image 11xray.png .

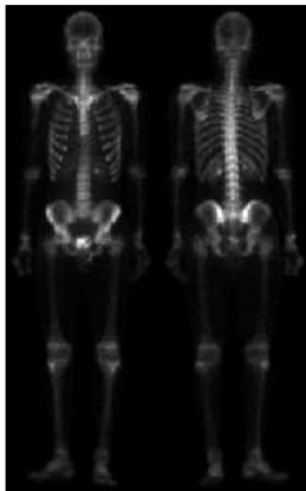
Apply the following techniques:

- (a) Apply Laplacian filter to the original image
- (b) Subtract the image from (a) from the original image to obtain a sharpened image
- (c) Apply both the horizontal and vertical Sobel filter to the image from (b)
- (d) Smooth image from (c) using a 5×5 averaging filter
- (e) Element-by-element multiply the images from (b) and (d)
- (f) Add the image from (e) to the original image to obtain a sharpened image
- (g) Apply power-law transformation (i.e., $g_0 = (g_i)^{0.5}$) to adjust the image contrast

Power-law transformation (a.k.a. gamma correction) is an approach to adjust the contrast of an image. In the transformation, the gray level of the pixels were adjust by the equation $g_0 = c(g_i)^r$, where g_0 is the gray level of the output image, g_i is the gray level of the input image, and c and r are pre-determined constants. The figure below explains the input-output relationship of power-law transformation. Note that in this example, we set $c = 1$.



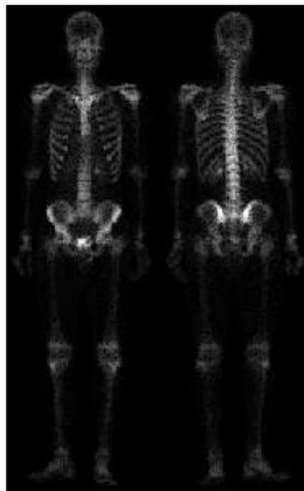
The images of each step are shown below.



Original image



(a)



(b)



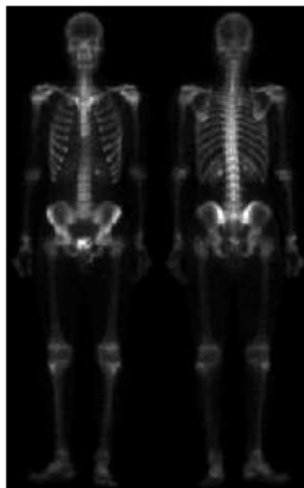
(c)



(d)



(e)



(f)



(g)

just do it

immultiply will make all the image white
it should divided 255 after multiply

```
img = imread('11Xray.png');  
img_a = imfilter(img, fspecial('laplacian'));  
img_b = img - img_a;  
img_c = img_b + imfilter(img_b, fspecial('sobel')) + imfilter(img_b, fspecial('average', 5));  
img_d = imfilter(img_c, fspecial('average', 5));  
img_e = uint8(double(img_b) .* double(img_d) / 255);  
img_f = uint8((double(img) + double(img_e)) / 2);
```

(g)

and the last one power-transform
it is easy and the effect is remarkable

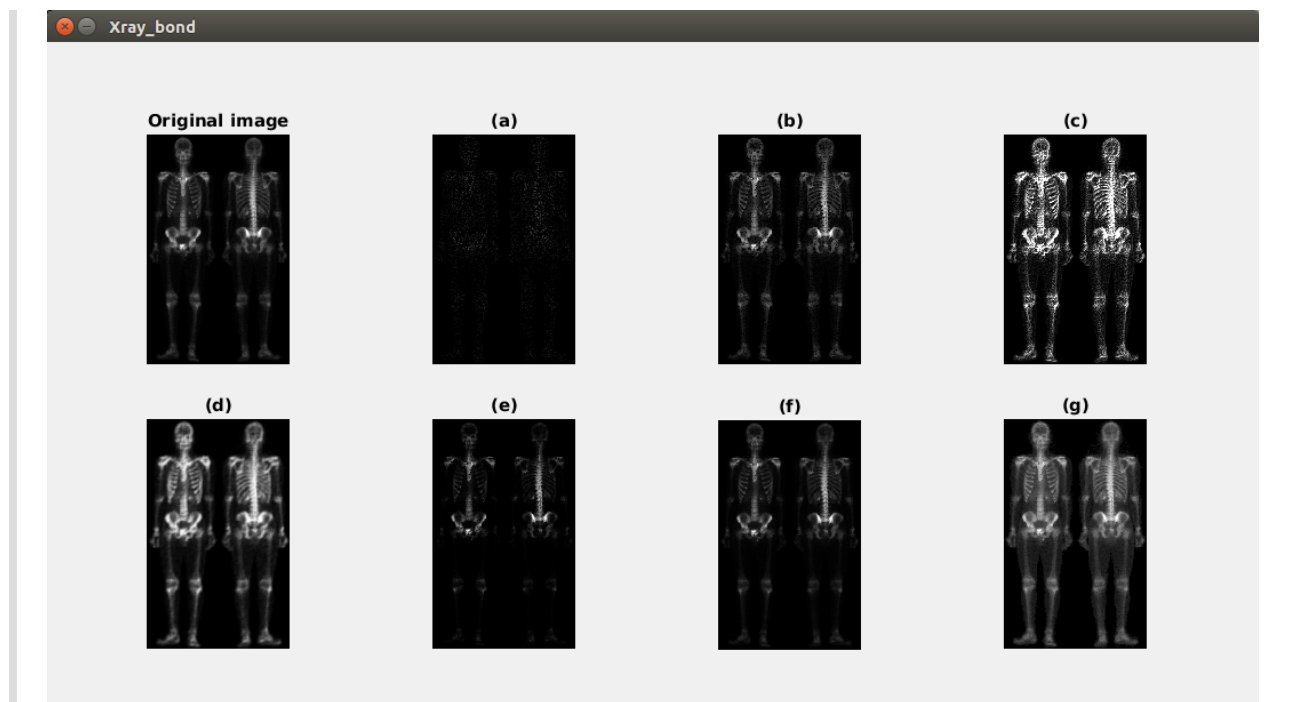
```
c = 1;  
g = 0.5;  
img_g = c * (double(img_f) / 255) .^ g;
```

i think that (e) and (f) can work is strange

final result

see enhancement_techniques.m
see Xray_bond.fig
see Xray_bond.m
see Xray_bond.png





star me on Github

linnil1 (https://github.com/linnil1/Lab304_2017summer)

Test on ubuntu16.04 + Matlab R2017a academic use