2110431 Introduction to Digital Imaging

2147329 Digital Image Processing and Vision Systems

Homework #1

Deadline : September 3rd, 2024 @23:59

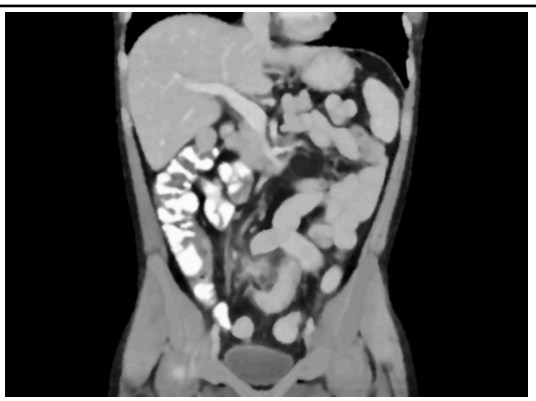
Submissions: (1) PDF version of this file

(2) .ipynb file; template in the link below

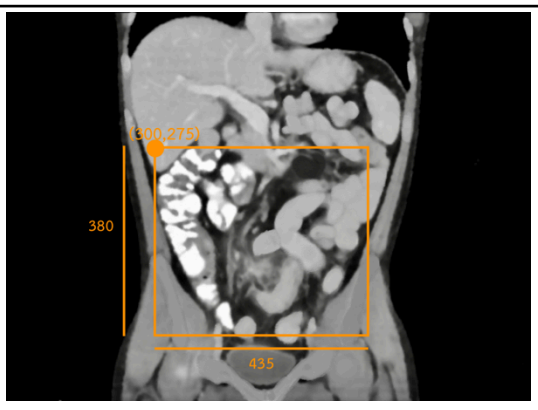
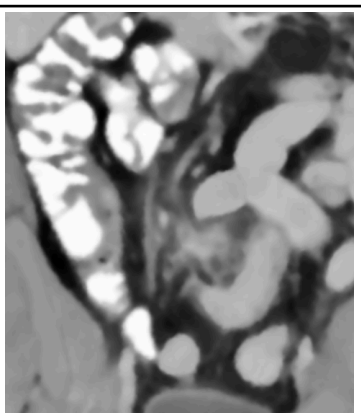
COLAB TEMPLATE: https://colab.research.google.com/drive/1pocG6rLwKfpp4wdFvH0kGT_AU36dwPxB?usp=sharing

1. Assume you work in the field of image processing. Your boss has assigned you a task to detect malignant tumors (assuming in this case, in bright intensity) from CT-SCAN images. The pain point is that the doctors saved images from the CT-SCAN, but the output images are incomplete and have Salt and Pepper noise. Please help the doctor remove the noise.

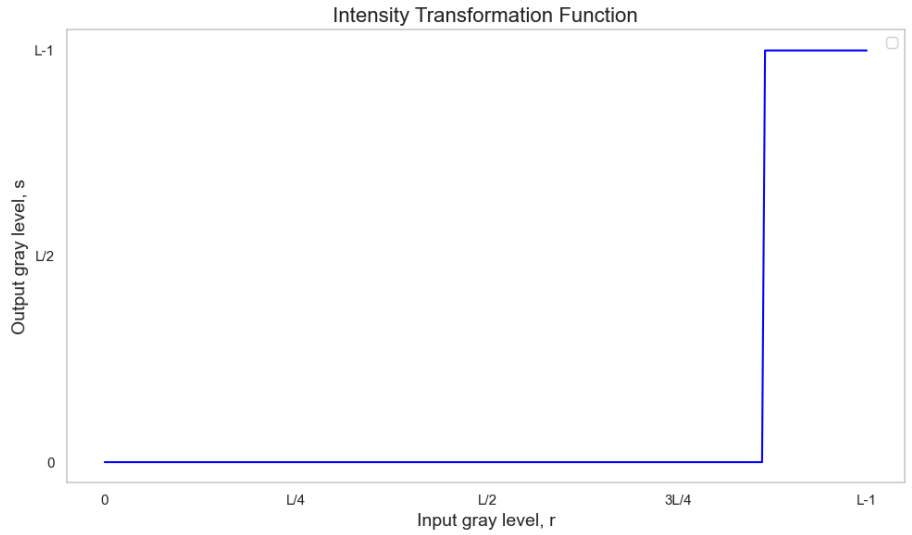
1.1 Apply a filter to remove the noise and select the appropriate size of the kernel. Provide your filtered image into the blank box below. Hint cv2.medianBlur

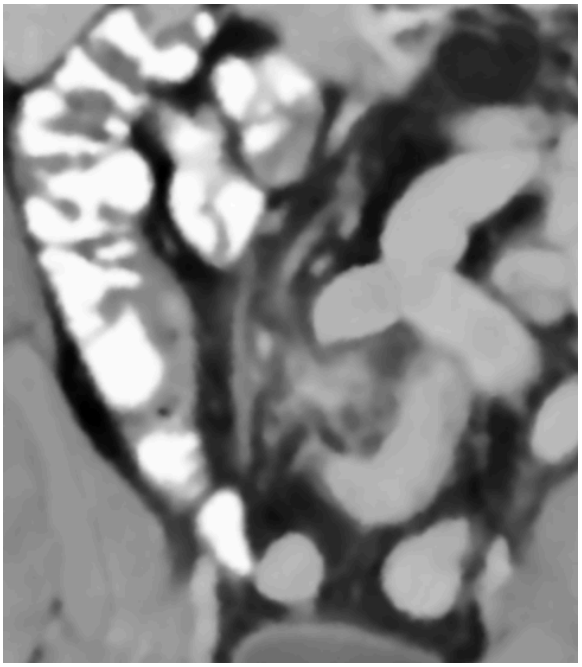
Original	Image result after Filtering
	

1.2 Apply Region of Interest (ROI) with width=380 and height=435 start at x=300, y=275 as shown in the orange rectangle below and provide the ROI image in the blank box below. Hint cv2.rectangle

Hint	Mark ROI Image
	

1.3 Apply the transformation function shown in the graph below on the ROI image. This transformation function is used for segmenting malignant tumors (assuming, in this case the higher intensity) and show in a white mask. Provide the final segmented tumors in the blank box below.

Transformation Function

L-1 is the highest intensity level.
You can estimate the stepping point from the graph.

Original (Mark ROI)	Segmentation with Transformation
	

You can write your own code or use the code template below and modify the _____ and put it in **`homework1_1()`** function in **`homework1.ipynb`** file

https://colab.research.google.com/drive/1394GXC-dHT7id5I6_xf8D5G2qFl8abVm?usp=sharing

2. Design your own filter on an RGB image. Write your code in `homework1_2()` function in **homework1.ipynb** file Provide motivation behind the designed filter. Display it in terms of an RGB image.

Idea / Motivation:

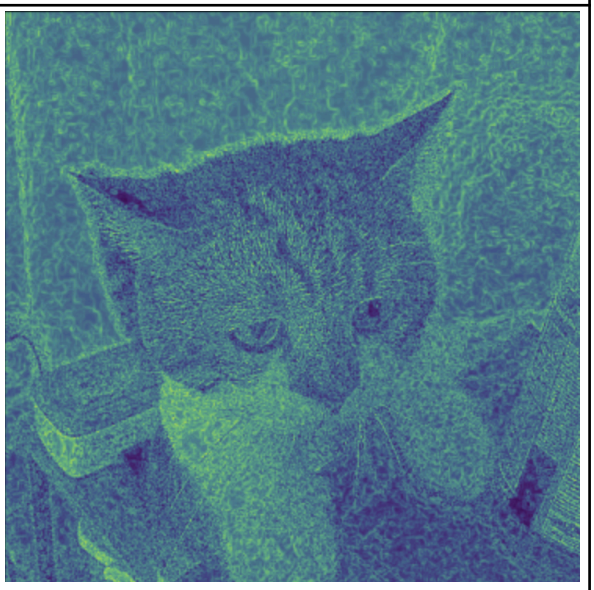
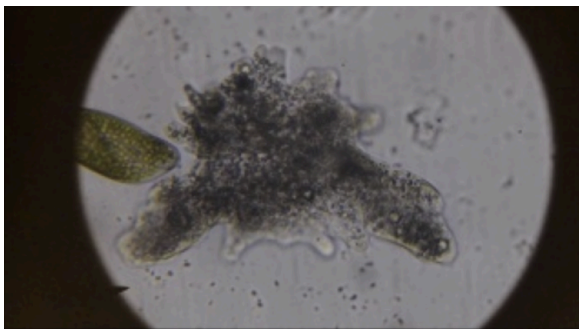
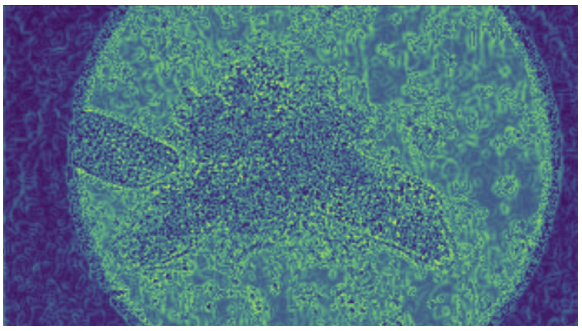
Finding edges of objects in the picture

Your filter design (at least two equations and/or conditions):

Sharpening kernel

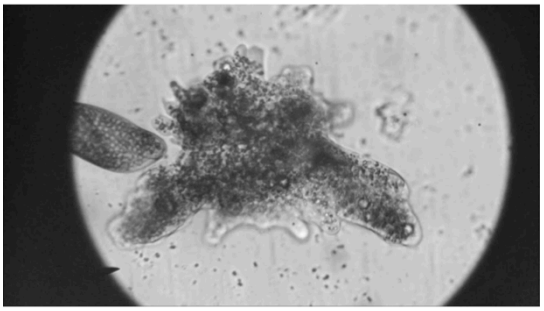
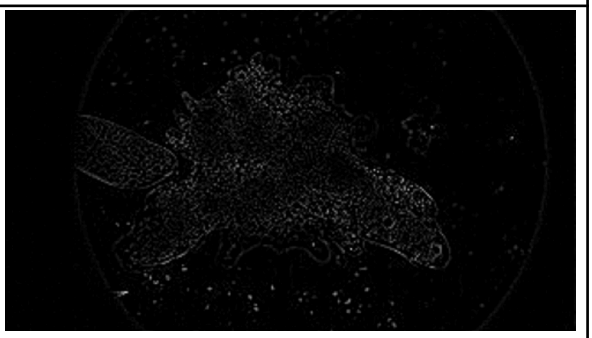
Sobel operator for edge detection

Examples of filtered image:

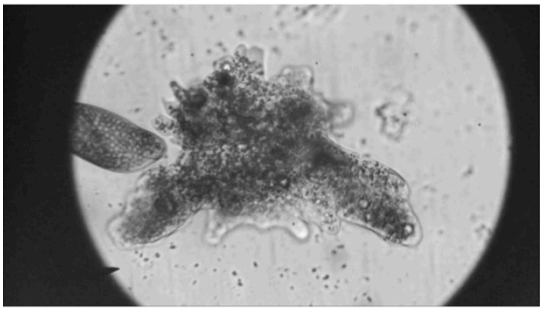
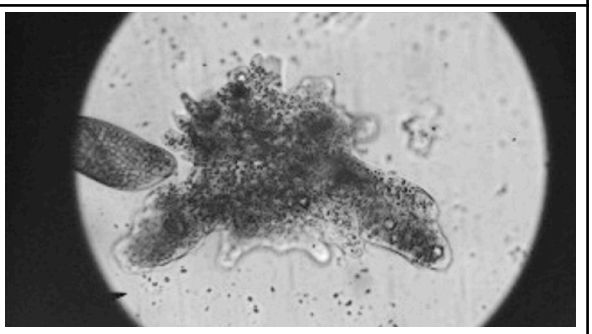
Original	After filtering
Kitty.jpg: 	
Your image: 	

3. Suppose you are a researcher in a lab. After obtaining an image of Amoeba, you want to enhance the sharpness of the image to study its various components. This can be achieved through sharpening spatial filtering using the Laplacian method. Write your code to show the Laplacian image and sharpening image in in `homework1_3()` function in **homework1.ipynb**

3.1 Apply a Laplacian filter and provide your resulted Laplacian image in the blank box below

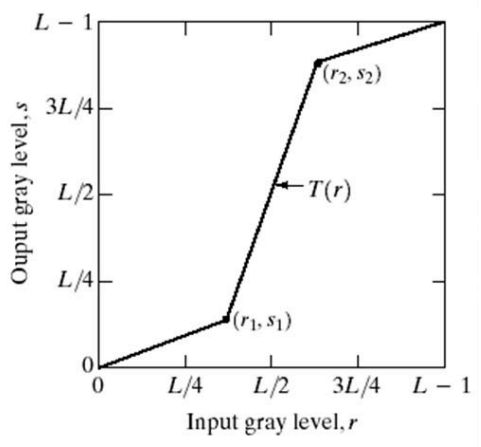
Original	Laplacian Image
amoeba.jpg 	
What is the Laplacian filter matrix you have used?	0 1 0 1 -4 1 0 1 0

3.2 Sharpening the image using the Laplacian filter in 3.1. Provide the result in the blank box below.
Hint: np.clip() helps you limit the values in an array

Original	Image after sharpening
amoeba.jpg 	
Explain briefly how to get the sharpened image.	Tiny details are more visible.

For practice

4. Write a python program to implement contrast stretching follow the transformation in the graph below.



Test your program using kitty.jpg and your own image and display your results in the blank below.

Results of the processed images:

Original	After filtering
Kitty.jpg	
Your image:	

5. Two images, `f(x,y)` and `g(x,y)`, have histograms h_f and h_g. Write a program to display the histograms h_f and h_g. Then implement the operations below and display the new histogram of the output of each operation. Determine the new histograms in terms of h_f and h_g and explain how to obtain the histogram in each case (Optional).

(a) `f(x,y) + g(x,y)`

(b) `f(x,y) - g(x,y)`

(c) `f(x,y) x g(x,y)`

(d) `f(x,y) ÷ g(x,y)`

Hint: design one of the images very simple.