In this lab, you will implement smoothing, sharpening and first derivative filters to grayscale images.

Important Note: You should complete the lab until the end of the lab hours and submit all your codes to SUCourse as a single zip file. Deadline for in-lab code submission to SUCourse is **17:00**.

Things to do:

Your functions must be as generic as possible, i.e., don't make any assumptions about the size, the type and the colors of the images. Your functions must convert the image to grayscale if it is colored and you must employ the row and column numbers of the images as variables.

- **Linear Filtering:** Gaussian filtering is a linear operation that involves the local convolution of a given image with a filter kernel of samples of the 2D Gauss function. Gaussian filtering is used for smoothing the pictures by eliminating "outliers" in image values which are considered to be noise in a given context. In order to apply Gaussian smoothing you can use the kernel below to convolve it with the given image.

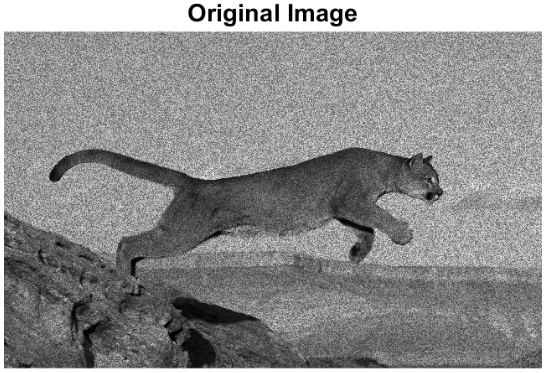
1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

/273

Now write a function which takes an image as input and returns the "**Gaussian smoothed**" version of the image. You will use 5×5 approximated Gaussian filter kernel given above. Your function name should be "lab2gaussfilt.m".

Your results should look as follows:

Original Image



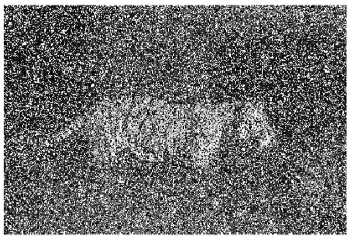
Gaussian Filtered Image

- **Nonlinear Filtering:** Median filtering is a nonlinear operation since it involves sorting and picking the median element. Due to the nonlinearity, median filter can not be implemented as a convolution.

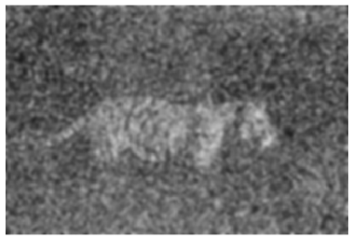
> Now write a function which takes an image and a constant k as inputs and returns the "**median filtered**" version of the image. Your function name should be "lab2medfilt.m".

Your results should look as follows:

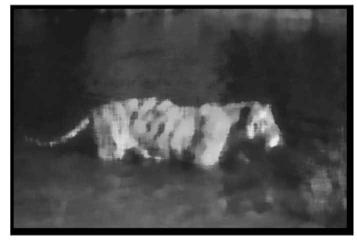
Original Image



Gaussian Filtered Image



Median Filtered Image



- **Sharpening:** The aim of the sharpening operation is to produce an enhanced image J by increasing the contrast of the given image I along edges, without adding too much noise within homogeneous regions in the image. Sharpening can be implemented as follows

$$J(p) = I(p) + \lambda\Big[I(p) - S(p)\Big] \tag{1}$$

where S is the smoothed version of the given image I and $\lambda > 0$ is a scaling factor which controls the influence of the correction signal.

> Now write a function which takes an image, a constant λ and an integer M as inputs and returns the "**sharpened**" version of the image. Use the integer M to select the method of smoothing such as "box filter", "Gaussian filter" and "median filter". Your function name should be "lab2sharpen.m".

Your results should look as follows:

Original Image

Sharpened Image

- **First Derivative:** *Sobel* filtering is a discrete 2D derivative operation which can be applied with the following kernels

-1	0	+1
-2	0	+2
-1	0	+1

x filter

+1	+2	+1
0	0	0
-1	-2	-1

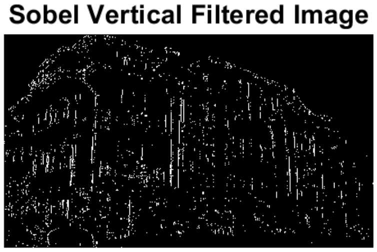
y filter

Now write a function which takes an image as input and utilize "**Sobel filter**" to return two images which are the first derivatives of the image along x and y directions. Your function name should be "lab2sobelfilt.m".

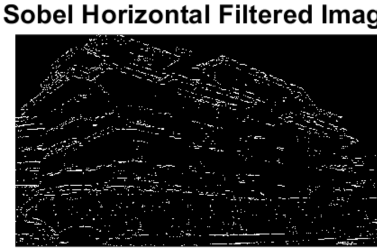
Your results should look as follows:

Original Image

Sobel Vertical Filtered Image



Sobel Horizontal Filtered Image



Post Lab

Post lab reports must include brief explanations of the functions that you implemented in this lab.

- Provide resulting images by utilizing all these functions. Comment on your results. Discuss the differences of the filters implemented in this lab in terms of their applicability for different scenarios.
- The 'sigma filter' is another nonlinear smoothing filter that suppresses the noise in a given image without significantly blurring it. Write a program that includes your own implementation of sigma filter and compare its results with the results of other filters.

Deadline for post lab report submission to SUCourse: **26 October 2021, 23:55**.