LAB # 05: Spatial Filtering

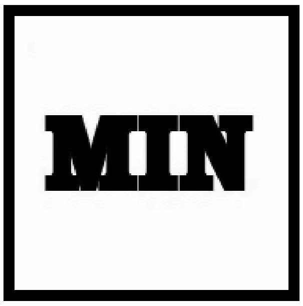
Objective:

The objective of this lab is to apply different spatial filters on images with different filter size.

Theory:

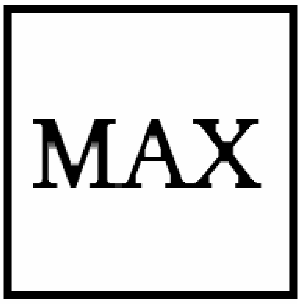
Different **spatial filters** allow us to enhance the details in an image as per our requirement. Median filters can help remove salt and pepper noise while applying max and min filter can affect the boundaries of different objects in an image. For example, when max filter is applied, the boundaries of dark objects will recede while the boundaries of light objects will expand. The effect of the min filter will be reversed. The effects of the aforementioned filters can be seen below:

Original Image

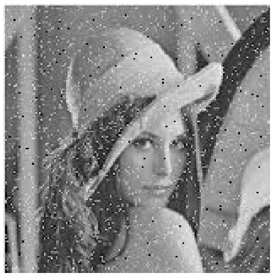


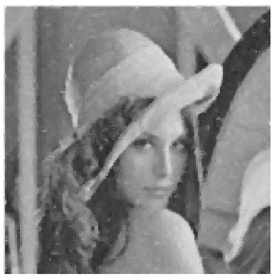
with Minimum Filter

Original Image



with Maximum Filter


Original Image


with Median Filter

In order to find out the boundary of objects in an image or edge detection, **Sobel** filters come in handy. Sobel filter calculates the first order derivative of the image. Horizontal Sobel filter is used to find out the edges along x-axis as it calculates the derivative of an image in x direction while Vertical Sobel filter is used to find out the edges along y-axis. The edges obtained from applying both these filters can then be added to obtain all the edges in an image. The result of Sobel filter ranges from -1020 to +1020. Where 0 stands for no change while -1020 and 1020 suggest maximum change from light to dark and dark to light respectively, but we are interested in the change not light to dark or dark to light change so to achieve that we can take absolute values only.

Applying the **Laplacian filter** gives the 2nd order derivative of the image. The edges can become more pronounced as compared to Sobel filter in some cases. The Laplacian is particularly sensitive to noise, so it is often used in combination with Gaussian smoothing (resulting in the Laplacian of Gaussian (LoG)).

Some Useful Commands:

1. To sort a list: my_list_sorted = **my_list.sort()** OR **numpy.sort(my_list)**
2. To get the median of the list: my_median = **numpy.median(my_list)**
3. To convert a n-dimensional matrix into a vector/1D array = **numpy.ravel(my_array)** OR **numpy.flatten(my_array)**

Lab Tasks:

Lab Task 1:

Apply the following filters with size 3, 15 and 31 on images Fig01.tif and Fig02.tif:

- Median
- Min
- Max

Lab Task 2:

Apply Horizontal Sobel and Vertical Sobel separately on the Fig03.tif. Display the results. Then add the images obtained by Horizontal Sobel and Vertical Sobel together and display the resultant images with all the edges.

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

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

The results can be combined in the following way where the Magnitude image is the image containing the magnitude and the Phase image contains the angels of the edges:

$$Magnitude = \sqrt{(Sobel_x)^2 + \left(Sobel_y\right)^2}$$

$$Phase = \tan^{-1}\left(\frac{Sobel_y}{Sobel_x}\right)$$

Do not forget to normalise the image before displaying the results.

Lab Task 3:

Apply the following Laplacian mask to image Fig03.tif. Then add the filtered image to the original image for a sharpening effect.

-1	-1	-1
-1	8	-1
-1	-1	-1

Conclusion:

This lab gives an understanding of applying different masks to images to obtain the desired result.

Home Task:

1. Take a high-quality grayscale image (e.g., a landscape or a document scan) and simulate degradation by introducing noise or blurring (you can apply a Gaussian blur or salt-and-pepper noise).
2. Apply histogram equalization to enhance the contrast of the degraded image.
3. Apply various filtering techniques (e.g., Gaussian filter, median filter) to reduce noise or sharpen the image. Use custom masks (e.g., 3x3 or 5x5) for both smoothing and edge enhancement.