Image Processing - Gradient

OpenCV provides three types of gradient filters or High-pass filters :-

- Sobel
- Scharr
- Laplacian

1. Sobel and Scharr Derivatives

Sobel operators is a joint Gaussian smoothing plus differentiation operation, which makes it even more resistant to noise. You can specify the direction of derivatives to be taken : either vertical or horizontal (by passing the arguments like *yorder* and *xorder*) .

- To specify the size of kernel : *ksize* argument
- if *ksize* = -1 : 3x3 Scharr filter is used
- if ksize != -1 : 3x3 Sobel filter is used.

Sobel Function in OpenCV

Code Structure :-

```
cv2.Sobel(src, ddepth, dx, dy, ksize)
```

- src : Image source object, read through imread function.
- ddepth : desired depth of the output image. Commonly used values are `CV_64F`, `CV_32F`, `CV_16S`, `CV_8U`
- dx : order of the derivative in the x-direction.
- dy : order of the derivative in the y-direction.
- ksize : The size of the sobel kernel used. It must be an odd number 1, 3, 5, 7. Kernel size affects the accuracy and smoothness of the gradient calculation. Larger kernel size relates to smoother gradients but may lose fine details.

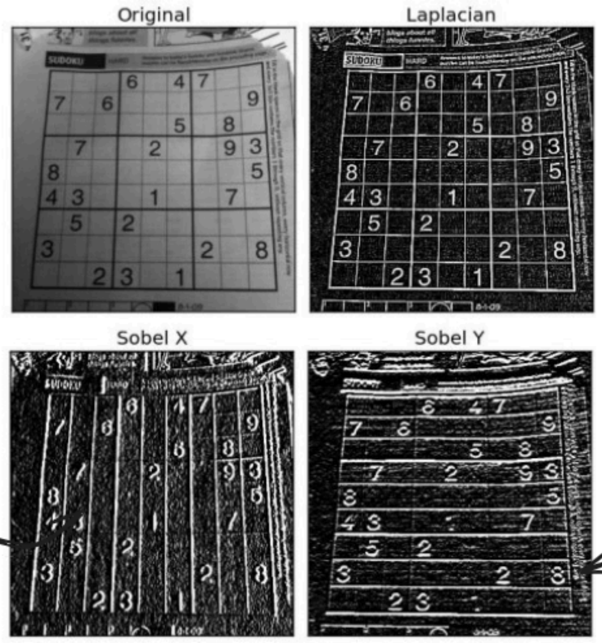
Scharr filter gives better results over Sobel filter.

2. Laplacian Derivatives

It calculates the laplacian of the image by the formula of laplace derivative, where each derivative is further solved using Sobel derivatives.

> **Tradeoff :**
> Since, laplacian method uses second-order derivative, it is more sensitive to any intensity changes, which is sometimes good but can also induce a lot of unnecessary noise in processed data. On the other hand, Sobel and scharr derivatives are of first order and hence work quite good enough for the task.

Notice, how laplacian compares to other methods



along x-axis

along y-axis

Edge detection algorithms

Different values for ksize(-1, 1, 3, 5, 7) generates different kernel matrix.

When using numpy as a library to process image edges, be careful. Since, different types of data may change slope direction. (Simple Calculus!)