National University of Sciences and Technology (NUST)
School of Electrical Engineering and Computer Science

Lab8: Edge Detection using spatial filtering

Objective:

The objective of this lab is:

- Understand the workings of standard edge detection filters (Sobel and Prewitt).
- Use standard edge detection filters to detect edges and analyze various tuning parameters.
- Implement Canny edge detector which uses hysteresis to detect edges efficiently.

Theory:

First Order Derivative kernels for Edge Detection:

Since edges can be described as sudden change in intensity values in some spatial direction, they can be detected by analyzing derivative of image. Unfortunately, a standard image is a discrete 2D light intensity function hence a finite difference approximations are usually used to estimate image gradients (as shown in following figure).

Derivative

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

Finite difference Approximation

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$

There exists various possibility to calculate aforesaid approximation with respect to x (i.e. backward, forward and central), for simplicity, we will be using central difference which can be calculated as:

$$f(x,y) = \frac{f(x+h,y) - f(x-h,y)}{2h} \Rightarrow$$

-1	0	1

x-derivative

$$f(x,y) = \frac{f(x,y+h) - f(x,y-h)}{2h} \Rightarrow$$

-1
0
1

y-derivative

Once image gradients in both x and y direction are calculated, both gradient values are usually combined in one way or the other to detect edges. For e.g. vital information (such as magnitude and direction of gradient) for each pixel can be calculated as follows:

Magnitude :

$$||\nabla f|| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

Direction :

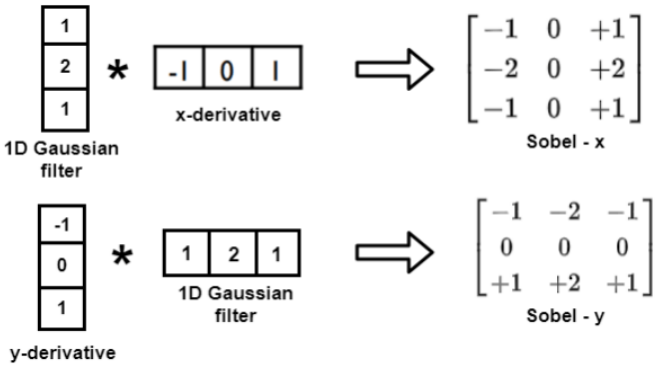
$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} \Big/ \frac{\partial f}{\partial x}\right)$$

Since edges are an abrupt change in the intensity values thus the largest gradient values will occur across an edge (neglecting noise) in an image. Thus, the x-gradient will find the vertical edges while y-gradient will highlight the horizontal edges as shown below.

Image | Gradient in x | Gradient in y

Edge detection with Sobel Operator:

Sobel operator is one of the basic technique to detect edges using the amount of change in either x or y gradient. Sobel's kernel for x and y gradient can be created by convolving 1D-Gaussian filter with either x or y 1D derivative operator as illustrated below:



Once, both Sobel-x and Sobel-y kernels are applied to the input image, their magnitude is then calculated:

$$M(x, y) \approx |g_x| + |g_y|$$

A threshold value can then be applied to M(x,y) to detect edges. Since the magnitude value contain information from both x and y gradients, both horizontal as well as vertical edges will be recorded by applying threshold value. Unfortunately, sobel edge detection process is rotation invariant. Similar to Sobel gradients, Prewitt kernel can also be employed for the calculation of magnitude.

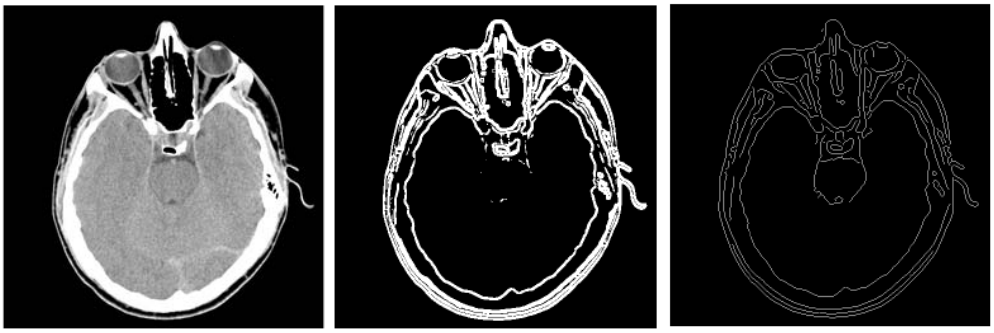
$$G_x = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \quad G_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

Prewitt kernel for image gradients.

Canny Edge Detector:

Unlike Sobel and Prewitt, The Canny edge detector uses a multi-stage algorithm to detect edges in images. It was developed by John F. Canny in 1986. Canny also produced a computational theory of edge detection explaining how the technique works. Steps involved in Canny edge detector are:

- Apply Gaussian filter to smooth out the image
- Find intensity gradients from the given image
- Apply non-maximum suppression to remove spurious response to edge detection
- Apply double threshold to determine potential edges
- Track edge by hysteresis: finalize the detection of edges by suppressing all the other edges that are weak and not connected to strong edges.



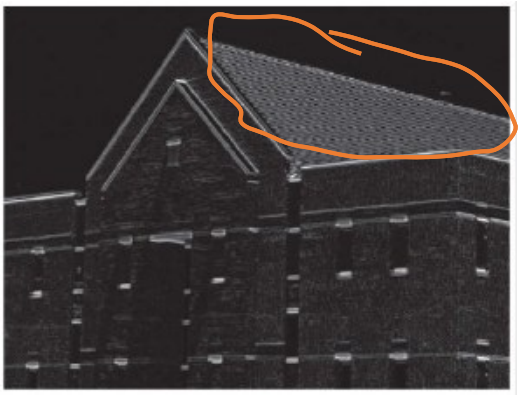
CT Image (left), thresholded gradient (middle) and edges detected using canny filter.

NOTE: Since implementation of canny filter require serious programming efforts, you can use opencv2's Canny() filter and experiment with low and high threshold parameters.

Lab Tasks:

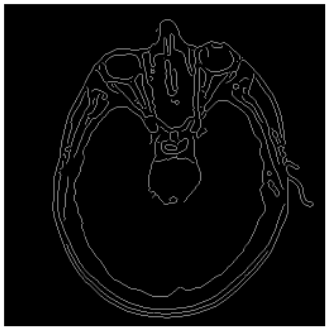
Task 1:

- Read "building.tif" image.
- Detect Edges using Sobel and Prewitt operators. (you should have two resulting images)
- Apply prior smoothing and then apply both Sobel and Prewitt operators. (now you have four images)
- Compare the results and report your findings.
- Try to detect edges of input image without following bricks. (HINT experiment with smoothing)



Task 2:

- Load “CT.tif” and apply edge detection using Sobel and Prewitt operator.
- Experiment with Low and High threshold of cv2.Canny() function and try to find optimal values which produce result similar to following.



- Similarly, apply Canny to “fingerPrint.tif” and try to detect edges in that image.
- Record and report your findings and analysis.

Submission Guidelines:-

Deliverables:

You have to submit a python notebook with ipynb extension containing all the codes and screen shots of output of all codes. Do not submit any zip folder containing code and screenshots separately.