Assignment Report

Canny Edge Detector Implementation

This report is for about assignment on the Canny edge detector, its implementation and results. The python libraries used for implementation are **Math, Matplotlib and Numpy. Basic functions of Numpy are used such as sqrt**.

First the images are read through Matplotlib. The ones in RGB or RGBA are converted to gray scale by standard gray scale conversion i.e. Y'=0.2627R'+ 0.6780G'+ 0.0593B'.

Image is blurred/smoothed with **Gaussian filter** of 5x5 to remove noise and get broader (spread) edges. In next step, we take **image gradient** with derivative filter. As derivatives are highly sensitive to noise, so blurring is an important preprocessing step. Gaussian filter is convolved with the image **without using any built-in functions for convolution**.

Masks are generated for sigma = 0.5, sigma=1, and sigma=2 with the given T =0.3 and first derivative of the Gaussian in X and Y direction i.e.

*fx = m.exp(-(x**2 + y**2)/(2*sigma**2)) * (x/(sigma**2))*

*fy = m.exp(-(x**2 + y**2)/(2*sigma**2)) * (y/(sigma**2))*

The values of fx and fy are **scaled up by a factor** of 255 and then rounded off to nearest integer so that convolution is performed with integers

Let say Gaussian kernel in X direction = Gx and in Y direction = Gy

Then for sigma=0.5 and T=0.3, Gx and Gy generated:

```
[[ -19.    0.   19.]   [[ -19. -138.  -19.]
 [-138.    0.  138.]    [   0.    0.    0.]
 [ -19.    0.   19.]]   [  19.  138.   19.]]
```

sigma=1 and T=0.3, Gx and Gy generated:

```
[[  -9.  -21.    0.   21.    9.]   [[  -9.  -42.  -69.  -42.   -9.]
 [ -42.  -94.    0.   94.   42.]    [ -21.  -94. -155.  -94.  -21.]
 [ -69. -155.    0.  155.   69.]    [   0.    0.    0.    0.    0.]
 [ -42.  -94.    0.   94.   42.]    [  21.   94.  155.   94.   21.]
 [  -9.  -21.    0.   21.    9.]]   [   9.   42.   69.   42.    9.]]
```

sigma=2 and T=0.3, Gx and Gy generated:

```
[[-20. -25. -18. 0.  18.  25.  20.] [[-20. -38. -55. -62. -55. -38. -20.]
 [-38. -47. -34. 0.  34.  47.  38.]  [-25. -47. -68. -77. -68. -47. -25.]
 [-55. -68. -50. 0.  50.  68.  55.]  [-18. -34. -50. -56. -50. -34. -18.]
 [-62. -77. -56. 0.  56.  77.  62.]  [  0.   0.   0.   0.   0.   0.   0.]
 [-55. -68. -50. 0.  50.  68.  55.]  [ 18.  34.  50.  56.  50.  34.  18.]
 [-38. -47. -34. 0.  34.  47.  38.]  [ 25.  47.  68.  77.  68.  47.  25.]
 [-20. -25. -18. 0.  18.  25.  20.]] [ 20.  38.  55.  62.  55.  38.  20.]]
```

Images are convolved with the kernels, as convolution requires flipping the mask in X direct ion and then in Y direction. But as Gaussian is symmetric in nature, hence we only need to fl ip Gx in Y direction and Gy in X direction.

The results of convolution are saved, for three sigma values, in the folders named after their respective image titles. Gx highlights the vertical edges while Gy highlights horizontal edges .

With kernel size of 3x3, edges appear sharper but as the kernel size increases to 5x5 or 7x7, edges become less sharp. This is because convolution is weighted summation of neighbors, hence information becomes smooth as we increase filter size.

Following are the results of ctscan image for sigma=0.5 and sigma=1 which clearly show the difference in intensity of edges detected in both the directions.

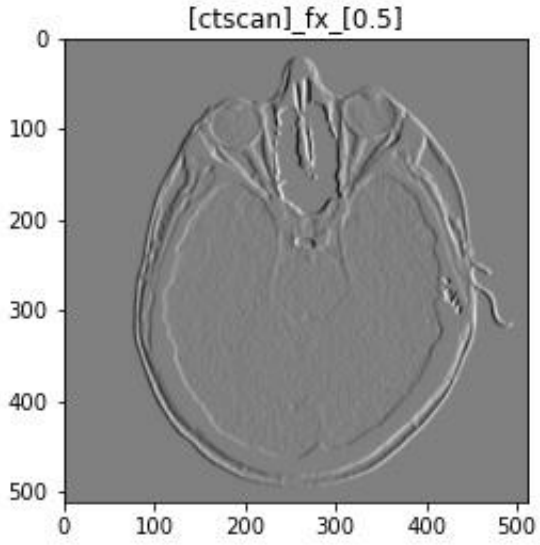


Figure 1

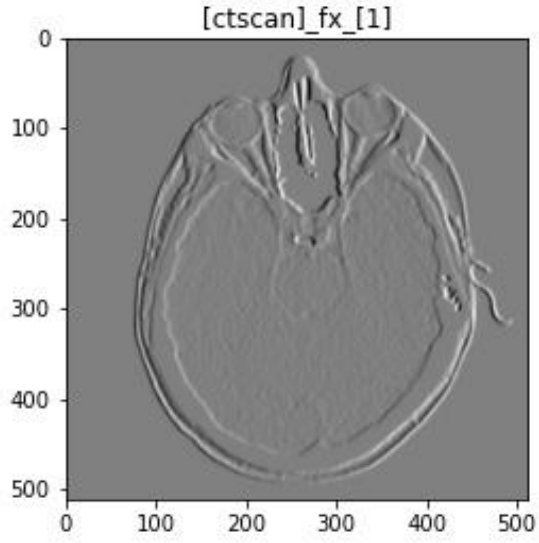


Figure 2

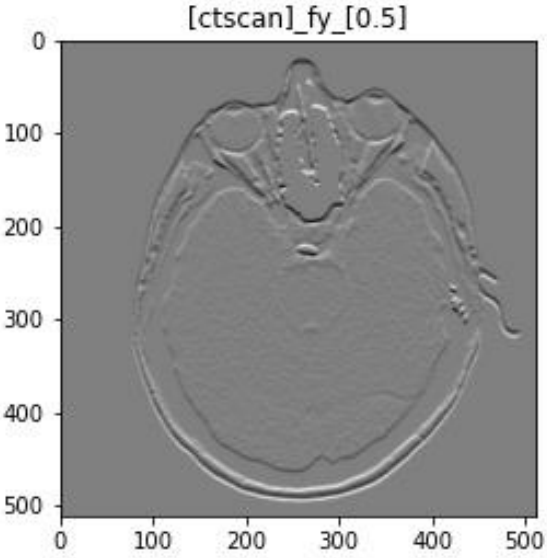


Figure 3

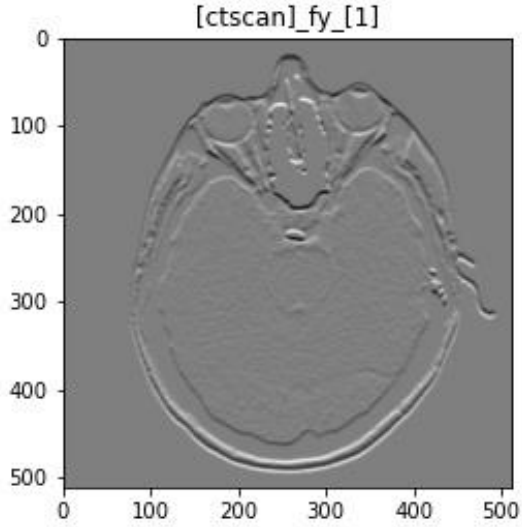


Figure 4

Next we **compute the Gradient magnitude** to get the maximum rate of change at a point. By the formula:

*np.sqrt(fx**2 + fy**2)/scale*

Where scale is used to scale down the values by same factor they were scaled up while gene rating masks, in our case 255. We get a higher intensity with greater sigma value but we ha ve to avoid noise as is the case with sigma=2, so sigma=1 value is better.

Below is the image of sigma=1.

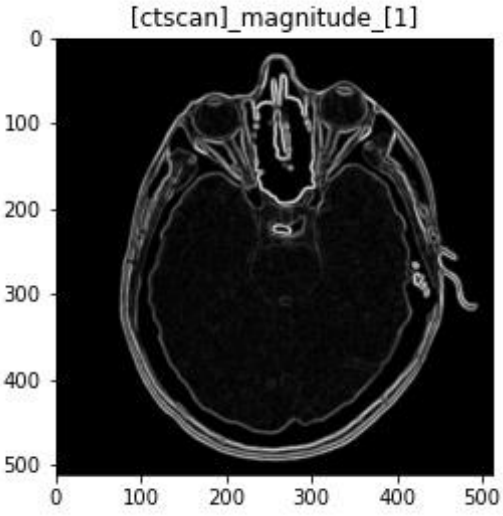


Figure 5

Next step is to compute the **direction of gradient** which will be used in the upcoming step to suppress the non-maximum edges. Magnitude direction is computed by:

*Gradient direction = (Arctan(fy/fx) *180/m.pi) +180*

The direction matrix is **further quantized to bins** such that angles in opposite direction are assigned same bin.

The color image of sigma=1 is as follows:

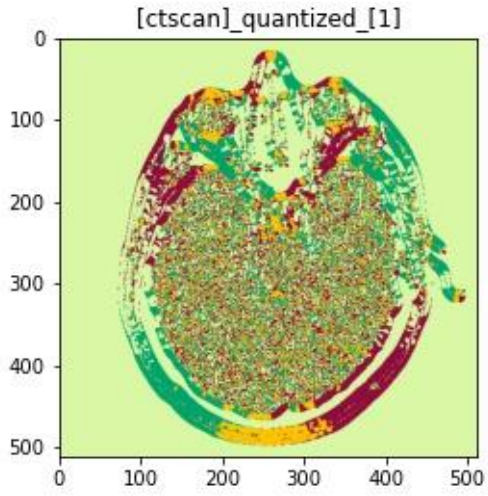


Figure 6

Next step is to perform **Non maximum suppression** over the gradient magnitude matrix M.

Take a value[I,j] in M and check for its direction in the quantized direction matrix. The value[I,j] is compared with its neighbors perpendicular to it, if value[I,j] is greater than both its perpendicular neighbors than it is maintained otherwise made zero. Neighbors with same bin values are the perpendicular neighbors and values are not altered in original matrix but rather stored to another matrix initialized with zeros.

Next step is to perform **Hysteresis thresholding** over the new non maximum suppressed matrix.

A value[I,j] if greater than a certain high threshold Th is considered to be an edge and all its neighbors if less than Th but greater than a low threshold T1 are also considered as edges. This process is done through recursion such that if a value[I,j] qualifies to be an edge then it is marked in a similar dimension matrix as visited and in another similar dimension matrix as edge. Its neighbors are then traversed and marked as visited, if any neighbor k qualifies to be an edge then neighbors of k are traversed and so on. A val[I,j] is allowed to visit if it has

not been visited before. The results of two different Th and T1 are shown for sigma=1. As Th and T1 increase, less edges are detected whereas with lower value, more edges are detected that might be noise.

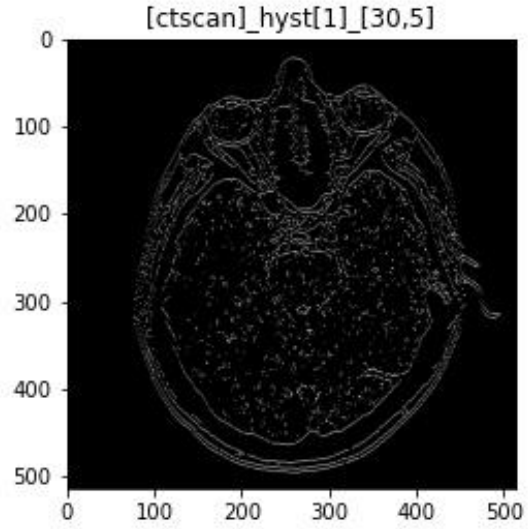


Figure 7

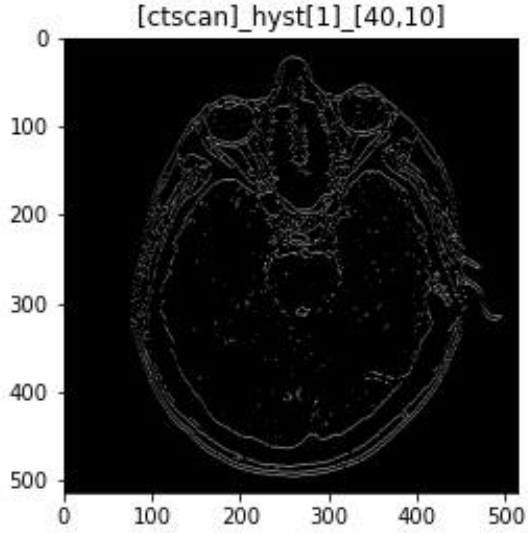
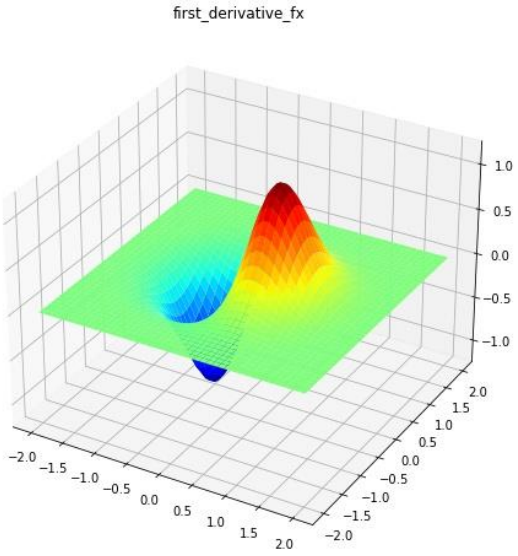


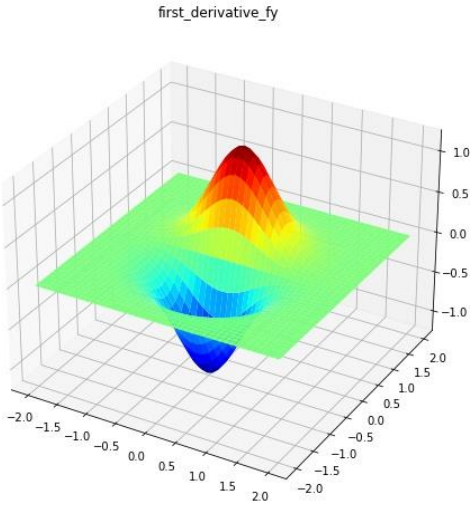
Figure 8

Mathematical Part1:

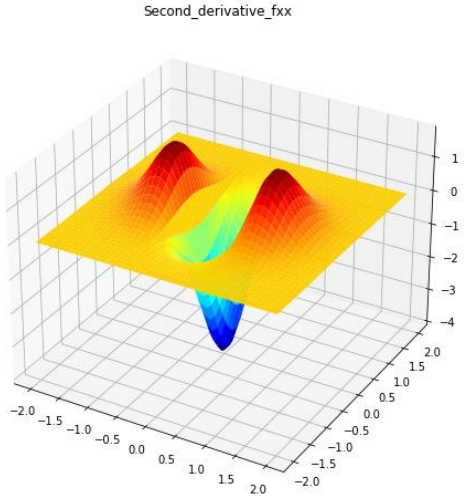
Plot of first derivative of Gaussian with respect to X direction



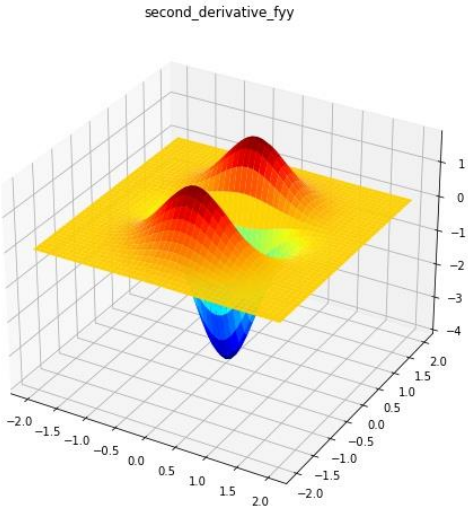
Plot of first derivative of Gaussian with respect to Y direction



Plot of Second derivative of Gaussian with respect to X direction



Plot of Second derivative of Gaussian with respect to Y direction



Plot of Laplacian of Gaussian by adding the second order derivatives both in X and Y direction