MACHINE LEARNING SMARTPENS

CODE CHALLENGE:Create a page or tool which performs edge detection on a given image and, given a point, returns the distance from that point to the closest edge.

1.USING OPENCV

OpenCV has an inbuilt function for performing canny edge detection.

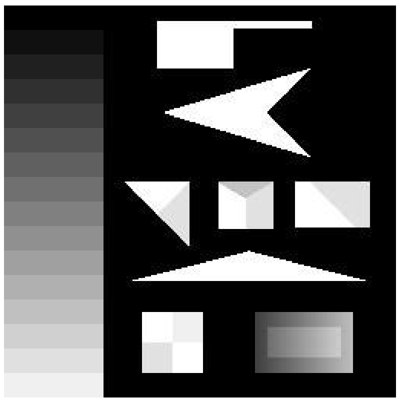
LinK:https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_canny/py_canny.html

This edge detection involves following steps:
i)Noise Reduction(using Gaussian filtering)
ii)Finding Intensity Gradient(using Sobel filters)
iii)Non Maximum Suppression
iv)Hysteresis Thresholding

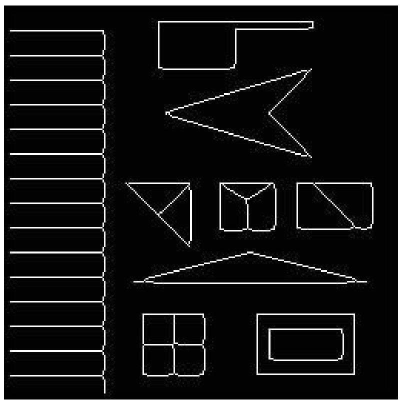
In the inbuilt function provided by OpenCV we can only modify the threshold range.We cannot make full use of this robust process from this inbuilt function itself.

Following limitations also come with this method:
-We sometimes need a measure of 'how much' the edge qualifies as an edge (e.g. intensity image coming from a Sobel amplitude edge detector)
-Also due to the gaussian smoothing: the location of the edges might be off, depending on the size of the gaussian kernel.
-The method has serious problems with corners and junctions:
-The gaussian smoothing blurs them out, making them harder to detect (same goes for the edges themself).
-The corner pixels look in the wrong directions for their neighbors, leaving open ended edges, and missing junctions.
These problems can be resolved by implementing own canny edge detection or using SUSAN detection for the same.

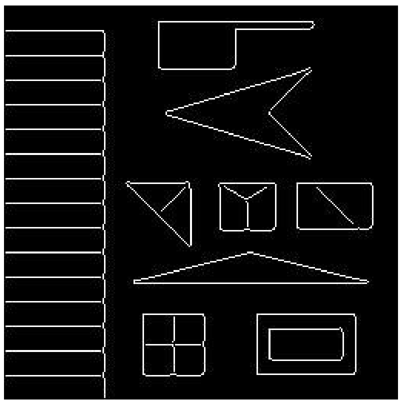
Test input image:



Results SUSAN:



Results Canny:



In the canny results we have missing pixels at corners for the above image.

Image link: (SUSAN) https://i.stack.imgur.com/FgXtr.png

(CANNY) https://i.stack.imgur.com/Q7ekN.png

I have also tried implementing canny edge detection so that we can customize it and specify parameters according to our requirement preventing any kind of edge loss.

canny.py is my implementation for the canny edge detection algorithm.I am still working on time complexity of the algorithm and I hope to come up with much better results in the upcoming week.

gsoc.py is the implementation for the code challenge. I have used breadth first search for finding the nearest edge using dequeue.