

Name – Akhil Dubey

Roll – 2020UCO1673

Branch – COE-3

Q1. Apply the prewitt and sobel filters on random image (that have fews edges), and compare the result with original image.

Q2. Increase the weight at positions [x,y+1], [x+1,y], [x-1,y], [x, y-1] in 3\*3 sobel filter, and infer how it will impact the edge image concerning the benchmark sobel.

Q3. Apply the LOG filter on the same image that you have taken in Q2, and compare the edge results with output of Q2.

import cv2

import numpy as np

from matplotlib import pyplot as plt

img = cv2.imread('nature.jpg',0) # 0 argument for grayscale

# Prewitt filter

prewitt\_x = np.array([[-1, 0, 1], [-1, 0, 1], [-1, 0, 1]])

prewitt\_y = np.array([[-1, -1, -1], [0, 0, 0], [1, 1, 1]])

# Sobel filter

sobel\_x = np.array([[-1, 0, 1], [-2, 0, 2], [-1, 0, 1]])

sobel\_y = np.array([[-1, -2, -1], [0, 0, 0], [1, 2, 1]])

# Prewitt

prewitt\_img\_x = cv2.filter2D(img, -1, prewitt\_x)

prewitt\_img\_y = cv2.filter2D(img, -1, prewitt\_y)

prewitt\_img = prewitt\_img\_x + prewitt\_img\_y

# Sobel

sobel\_img\_x = cv2.filter2D(img, -1, sobel\_x)

sobel\_img\_y = cv2.filter2D(img, -1, sobel\_y)

sobel\_img = sobel\_img\_x + sobel\_img\_y

plt.subplot(2, 2, 1), plt.imshow(img, cmap='gray')

plt.title('Original'), plt.xticks([]), plt.yticks([])

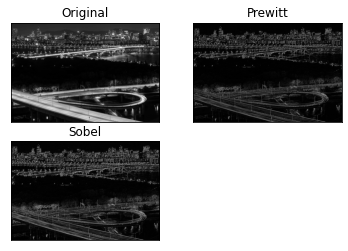
plt.subplot(2, 2, 2), plt.imshow(prewitt\_img, cmap='gray')

plt.title('Prewitt'), plt.xticks([]), plt.yticks([])

plt.subplot(2, 2, 3), plt.imshow(sobel\_img, cmap='gray')

plt.title('Sobel'), plt.xticks([]), plt.yticks([])

plt.show()



# Modified Sobel filter

sobel\_x\_mod = np.array([[-1, 0, 1], [-3, 0, 3], [-1, 0, 1]])

sobel\_y\_mod = np.array([[-1, -3, -1], [0, 0, 0], [1, 3, 1]])

sobel\_img\_mod\_sum = sobel\_x\_mod + sobel\_y\_mod

cv2.imshow('original image',sobel\_img\_mod\_sum)

# LoG filter

ksize = 5 # kernel size

sigma = 1.4 # standard deviation

kernel = cv2.getGaussianKernel(ksize, sigma)

kernel = kernel \* kernel.T

kernel = cv2.filter2D(kernel, -1, np.array([[-1, -1, -1], [-1, 8, -1], [-1, -1, -1]]))

log\_img = cv2.filter2D(img, -1, kernel)

plt.subplot(1, 2, 1), plt.imshow(img, cmap='gray')

plt.title('Original'), plt.xticks([]), plt.yticks([])

plt.subplot(1, 2, 2), plt.imshow(log\_img, cmap='gray')

plt.title('LoG Filter'), plt.xticks([]), plt.yticks([])

plt.show()

