CS – 532 – A

Janmejay Mohanty

Homework 4

Source Code:

```
import numpy as np
import open3d as o3d
from IPython.display import Image
import cv2
import math
def Gaussian_func(image,weight):
    sigma = (weight-1)/6
    gauss = np.zeros((weight,weight))
    for i in range(weight):
        for j in range(weight):
            gauss[i,j] = math.exp(-((i)**2+(j)**2)/(2*sigma**2))/(2*math.pi*sigma**2)
    return cv2.filter2D(image,-1,gauss)
def Suppression_func(image):
    height,breadth = image.shape
    result = np.zeros((height,breadth))
    for i in range(2,(height - 1)):
        for j in range(2,(breadth - 1)):
            suppress = image[i-1:i+1, j-1:j+1]
            if(image[i,j] == suppress.max()):
                result[i,j] = image[i,j]
    return result
Dict_depth = {}
Dict_rgb = {}
Dict_gauss = {}
for i in range(3):
    Dict_depth[i] = cv2.imread("problem1/depth" + str(i+1) + ".png")
```

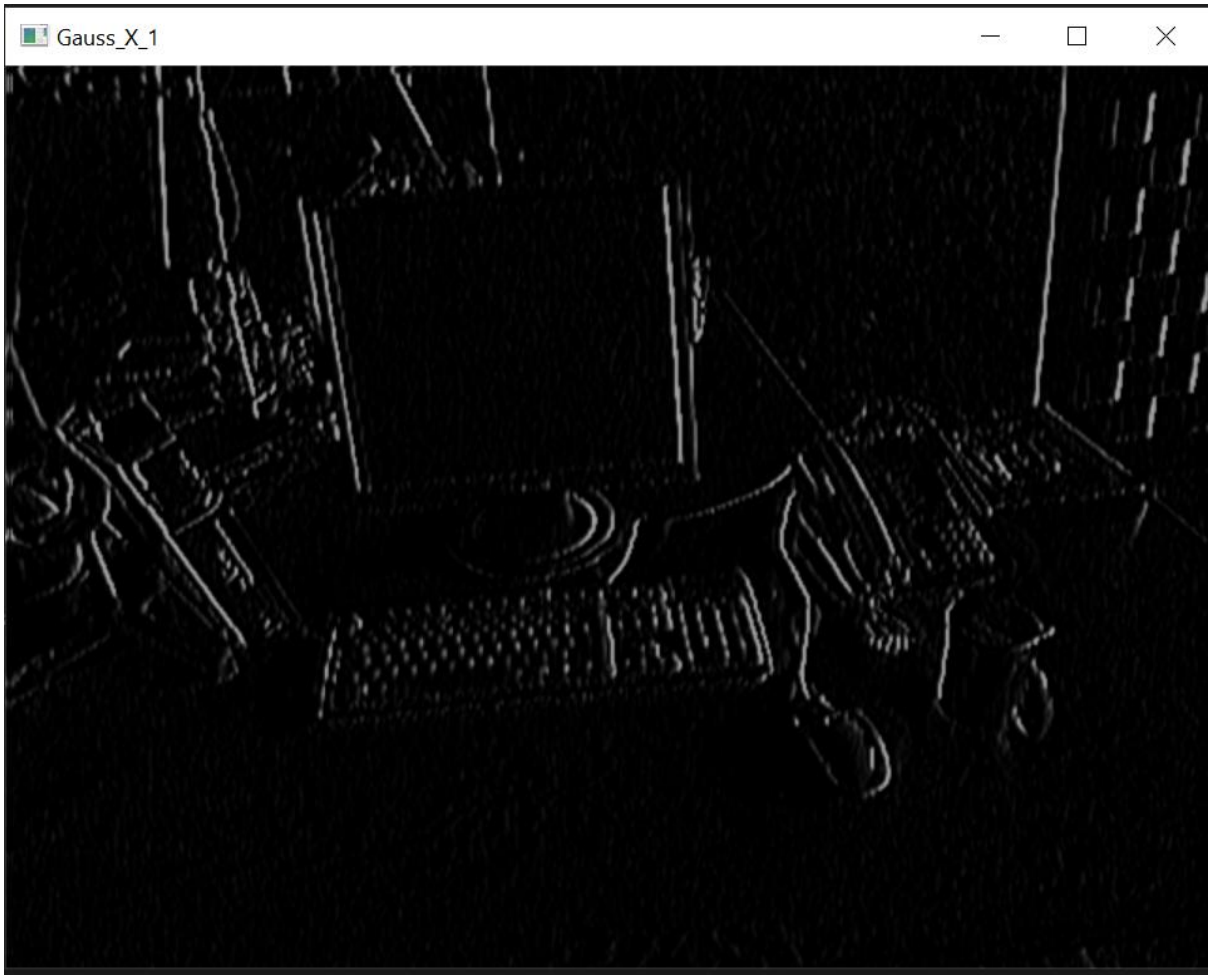
```
    Dict_rgb[i] = cv2.imread("problem1/rgb" + str(i+1) + ".png")
    Dict_gauss[i] = cv2.cvtColor(Dict_rgb[i], cv2.COLOR_BGR2GRAY)
Kernel_x = np.array([[-1,0,1],[-1,0,1],[-1,0,1]])
Kernel_y = np.transpose(Kernel_x)
print(Kernel_y)
x_1 = cv2.filter2D(Dict_gauss[0],-1,Kernel_x)
x_2 = cv2.filter2D(Dict_gauss[1],-1,Kernel_x)
x_3 = cv2.filter2D(Dict_gauss[2],-1,Kernel_x)

y_1 = cv2.filter2D(Dict_gauss[0],-1,Kernel_y)
y_2 = cv2.filter2D(Dict_gauss[1],-1,Kernel_y)
y_3 = cv2.filter2D(Dict_gauss[2],-1,Kernel_y)

X_1 = cv2.filter2D(x_1,-1,Kernel_x)
X_2 = cv2.filter2D(x_2,-1,Kernel_x)
X_3 = cv2.filter2D(x_3,-1,Kernel_x)

XY_1 = cv2.filter2D(y_1,-1,Kernel_x)
XY_2 = cv2.filter2D(y_2,-1,Kernel_x)
XY_3 = cv2.filter2D(y_3,-1,Kernel_x)

Y_1 = cv2.filter2D(y_1,-1,Kernel_y)
Y_2 = cv2.filter2D(y_2,-1,Kernel_y)
Y_3 = cv2.filter2D(y_3,-1,Kernel_y)
# Apply Gaussian smoothing to the derivatives using the 5x5 filter
Gauss_X_1 = Gaussian_func(X_1,5)
Gauss_X_2 = Gaussian_func(X_2,5)
Gauss_X_3 = Gaussian_func(X_3,5)

Gauss_XY_1 = Gaussian_func(XY_1,5)
Gauss_XY_2 = Gaussian_func(XY_2,5)
```

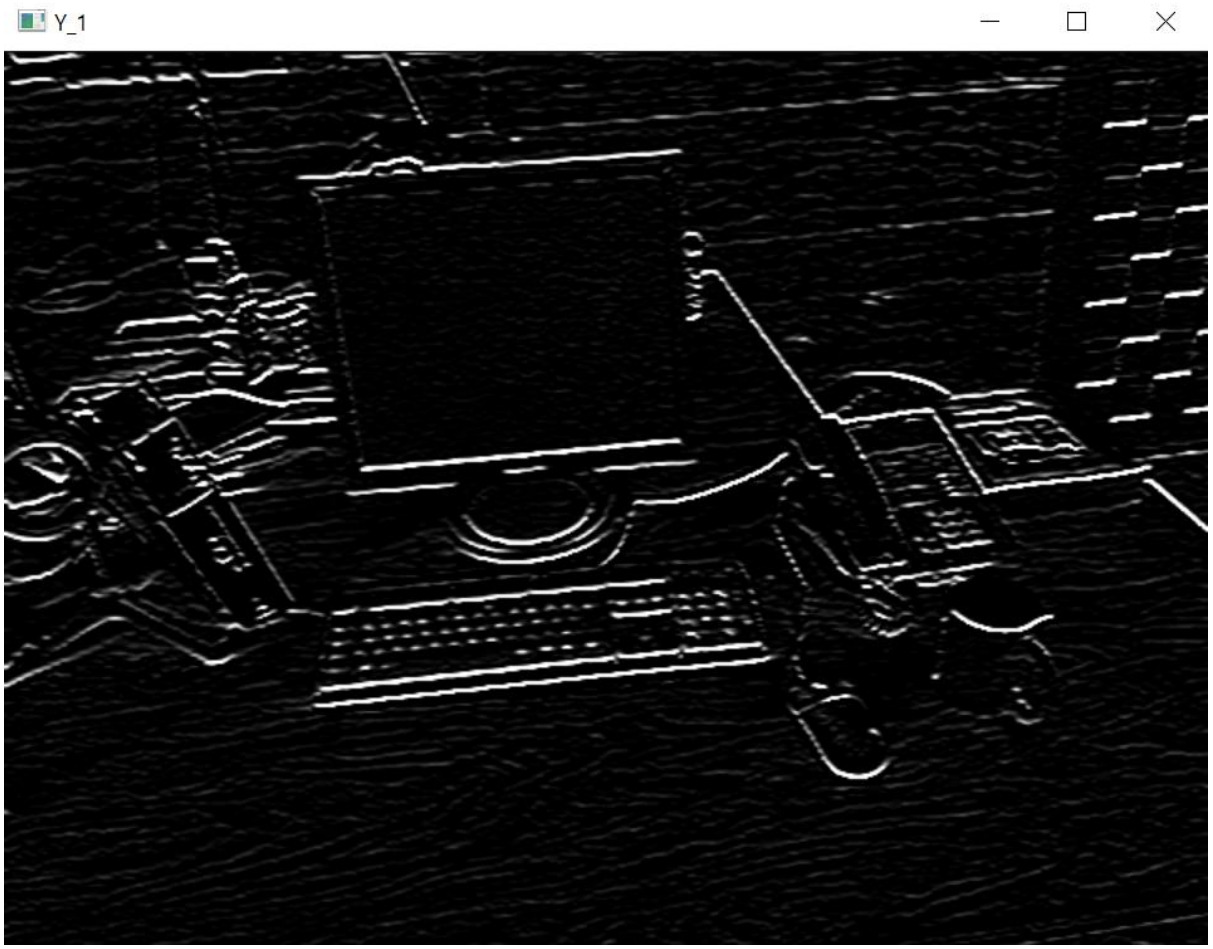
```
Gauss_XY_3 = Gaussian_func(XY_3,5)


Gauss_Y_1 = Gaussian_func(Y_1,5)

Gauss_Y_2 = Gaussian_func(Y_2,5)

Gauss_Y_3 = Gaussian_func(Y_3,5)

cv2.imshow("X_1", X_1)

cv2.imshow("XY_1", XY_1)

cv2.imshow("Y_1", Y_1)

cv2.imshow("Gauss_X_1", Gauss_X_1)

cv2.imshow("Gauss_XY_1", Gauss_XY_1)

cv2.imshow("Gauss_Y_1", Gauss_Y_1)

cv2.waitKey(0)

a = 0.05

mr = np.multiply(Gauss_X_1,Gauss_Y_1)

mr = (np.multiply(Gauss_X_1,Gauss_Y_1) - np.square(Gauss_XY_1,Gauss_XY_1)) -
a*np.square(Gauss_X_1+Gauss_Y_1)

cv2.imshow("Rendered Image", mr)

cv2.waitKey(0)

sup = Suppression_func(mr)

cv2.imshow("Supressed Image", sup)

cv2.waitKey(0)
```

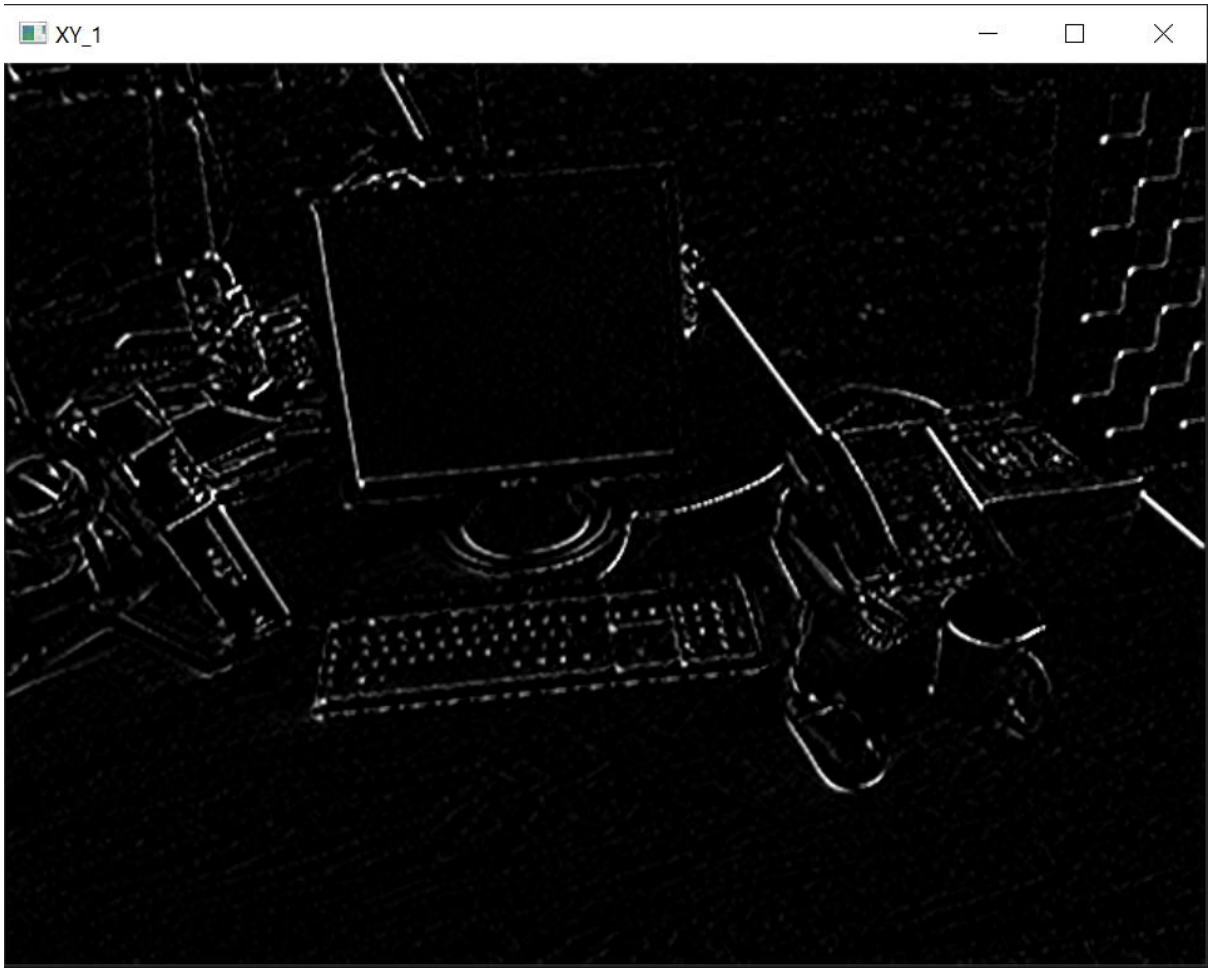
Output Images:

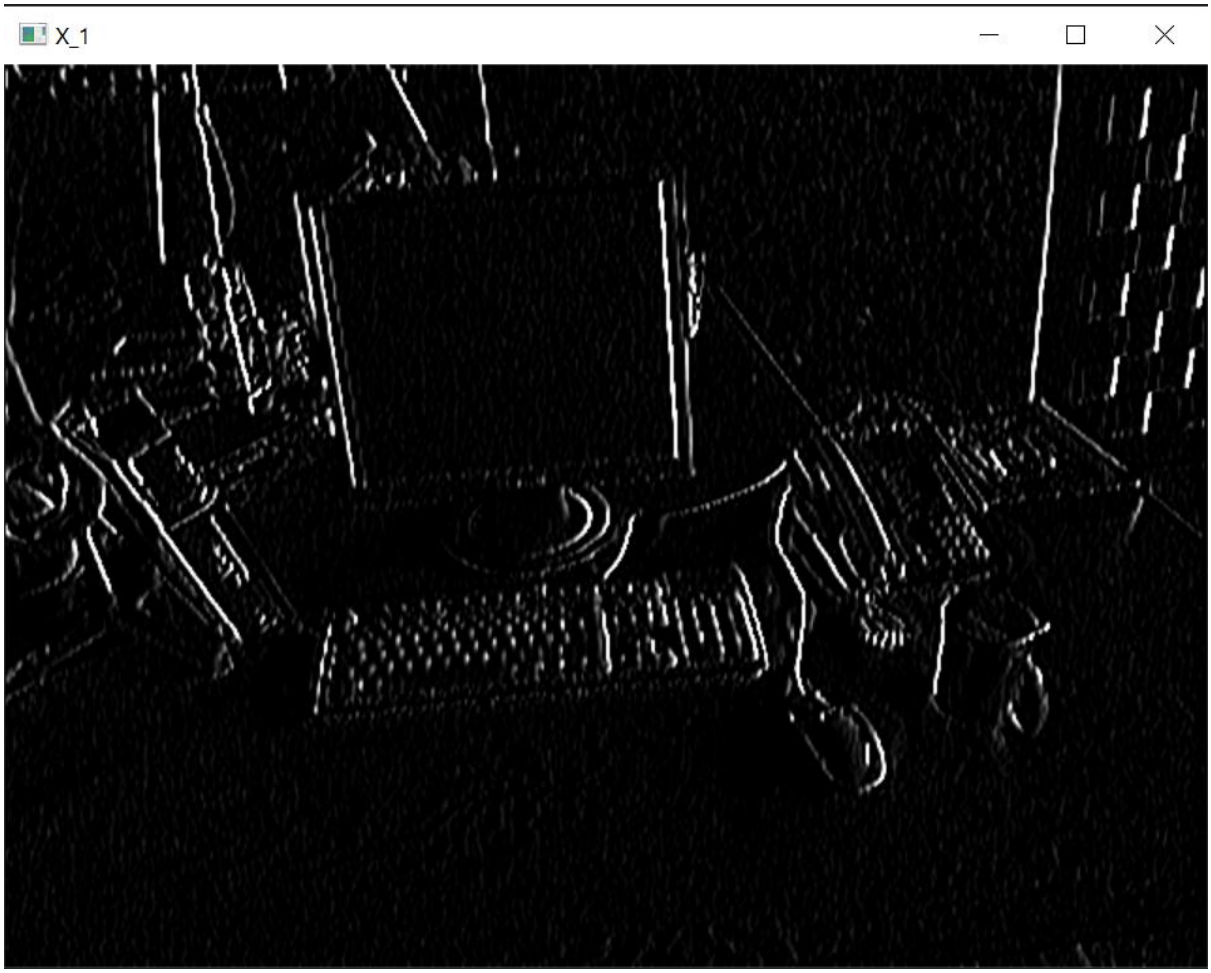
```
...   [[-1 -1 -1]
       [ 0  0  0]
       [ 1  1  1]]
```

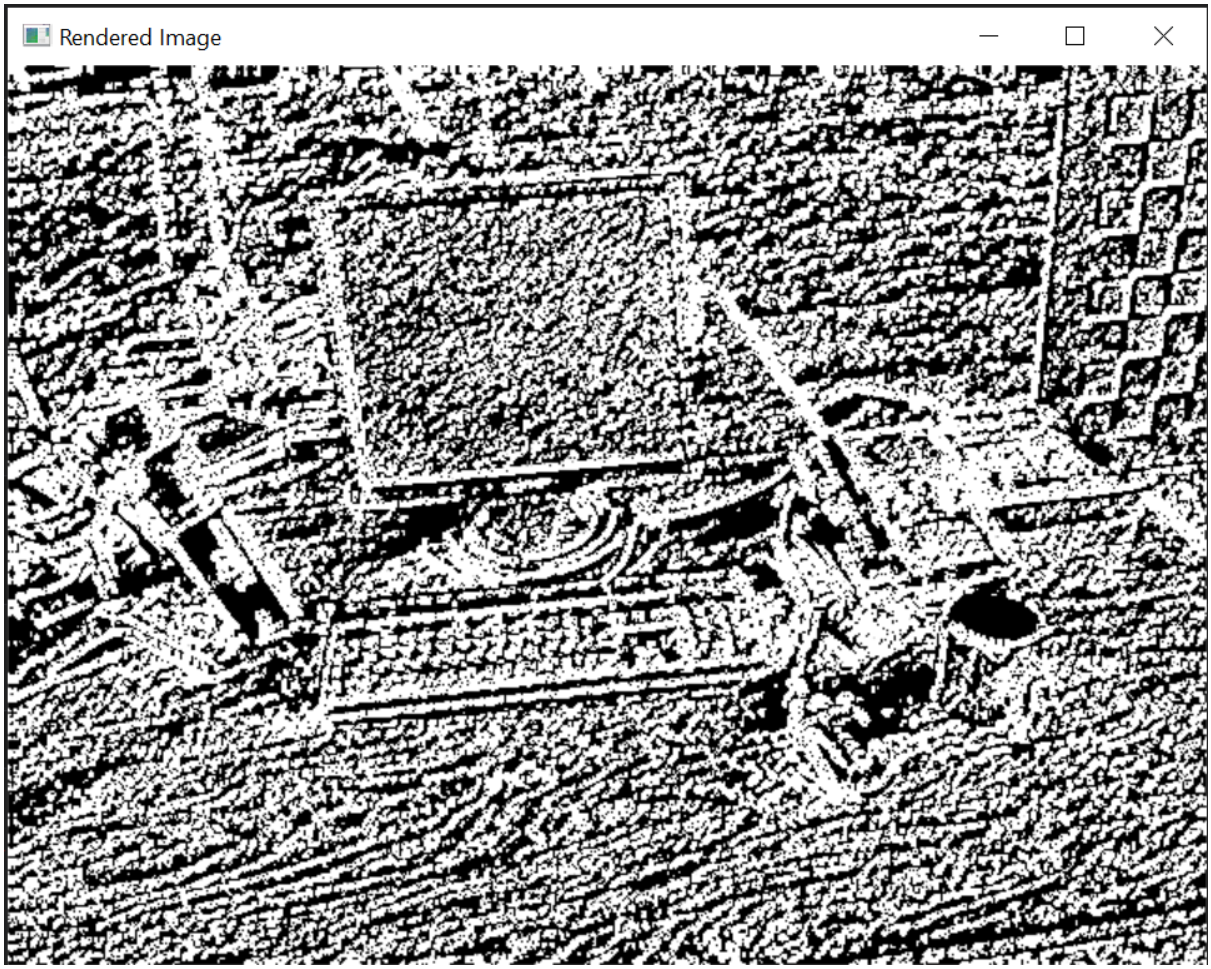

Gauss_XY_1


Gauss_X_1


Y_1


XY_1


X_1


Rendered Image


Supressed Image