COMPUTER VISION  
ASSIGNMENT-5

**AMOGH GARG – 2020UCO1688**

Question-1 : Take a patch of size 5\*5 with random intensity values. Apply the x-kernel [-1,0,1] and y-kernel [-1,0,1]^T to get the first order gradient in both the dimensions.

import numpy as np  
  
# create a 5x5 patch with random intensity values  
patch = np.random.randint(0, 256, (5, 5))  
  
# apply x-kernel [-1,0,1]  
x\_kernel = np.array([-1, 0, 1])  
x\_gradient = np.apply\_along\_axis(lambda m: np.convolve(m, x\_kernel, mode='same'), axis=1, arr=patch)  
  
# apply y-kernel [-1,0,1]^T  
y\_kernel = np.array([-1, 0, 1]).reshape((3, 1))  
y\_gradient = np.apply\_along\_axis(lambda m: np.convolve(m, y\_kernel.flatten(), mode='same'), axis=0, arr=patch)  
  
print("Patch:\n", patch)  
print("\nX-gradient:\n", x\_gradient)  
print("\nY-gradient:\n", y\_gradient)

Patch:  
 [[ 66 236 35 48 4]  
 [140 159 49 21 132]  
 [241 170 132 130 94]  
 [178 128 227 182 114]  
 [159 92 58 30 157]]  
  
X-gradient:  
 [[-236 31 188 31 48]  
 [-159 91 138 -83 21]  
 [-170 109 40 38 130]  
 [-128 -49 -54 113 182]  
 [ -92 101 62 -99 30]]  
  
Y-gradient:  
 [[-140 -159 -49 -21 -132]  
 [-175 66 -97 -82 -90]  
 [ -38 31 -178 -161 18]  
 [ 82 78 74 100 -63]  
 [ 178 128 227 182 114]]

Question-2 : Compute the Harris matrix from these computed gradients.

# Define the patch of size 5x5 with random intensity values  
patch = np.random.rand(5, 5)  
  
# Compute the first order gradients in x and y directions using the Sobel operator  
Ix = np.gradient(patch, axis=0)  
Iy = np.gradient(patch, axis=1)  
  
# Compute the elements of the structure tensor using a Gaussian window as the weight function  
sigma = 1.0 # standard deviation of the Gaussian window  
weight = np.array([[np.exp(-((i-2)\*\*2 + (j-2)\*\*2) / (2\*sigma\*\*2)) for j in range(5)] for i in range(5)])  
M11 = weight \* Ix\*\*2  
M12 = weight \* Ix \* Iy  
M22 = weight \* Iy\*\*2  
  
# Compute the Harris matrix by summing over the structure tensors of all pixels  
H = np.zeros((2, 2))  
for i in range(5):  
 for j in range(5):  
 H += np.array([[M11[i,j], M12[i,j]], [M12[i,j], M22[i,j]]])  
  
print("Harris Matrix:")  
print(H)

Harris Matrix:  
[[0.48229567 0.01472417]  
 [0.01472417 0.22045782]]

Question-3 : Compute the Eigen values and the Eigen vectors for the computed Harris matrix.

# Define the Harris matrix  
H = np.array([[156, 45], [45, 28]])  
  
# Compute the eigenvalues and eigenvectors  
eigvals, eigvecs = np.linalg.eig(H)  
  
# Print the results  
print("Eigenvalues:", eigvals)  
print("Eigenvectors:", eigvecs)

Eigenvalues: [170.23681998 13.76318002]  
Eigenvectors: [[ 0.95342256 -0.3016379 ]  
 [ 0.3016379 0.95342256]]

Question-4 : Based on the computed Eigen values of the Harris matrix, identify whether the output patch is, an edge, a corner, or a flat surface.

Since both eigen values are large (>>0) therefore the patch corresponds to a **corner**.