Goncalo Mendes

Computer Vision – Assignment 3

In this assignment, the handouts for the Canny Edge detection and Harris Corner Detection algorithms were read and the mathematical background behind it was understood.

Furthermore, to see the results on some of the sample pictures, through the OpenCV library we were able to test both algorithms with Python. The parameters passed when calling the OpenCV library function were searched online when finding more about the algorithms. For example, the image has to be converted to Grayscale in order to call the function.

Finally, the code was translated to C# and with the “Emgu CV” library, we were able to test both algorithms in this environment as well.

With the Power Point given, it was a good exercise to go over the math in the background of the Canny Edge Detection algorithm.

For Python, ahead is the source code and the examples for the two images.

import numpy as np

import cv2

from matplotlib import pyplot as plt

#Load a color image in grayscale

img = cv2.imread('C:/Users/Windows User/Desktop/UB Spring 2019/Computer Vision/Assignment 2/Sample\_Image.png',1) # 0 for gray, 1 for color

cv2.imshow('image',img)

#---------------canny edge detection-------------

img = cv2.imread('C:/Users/Windows User/Desktop/UB Spring 2019/Computer Vision/Assignment 2/Sample\_Image.png',0)

edges = cv2.Canny(img,50,200)

plt.subplot(121),plt.imshow(img,cmap = 'gray')

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

plt.subplot(122),plt.imshow(edges,cmap = 'gray')

plt.title('Edge Image'), plt.xticks([]), plt.yticks([])

plt.show()

#-------------------------------------------------

#--------------------Harris Corner Detection------

filename = 'C:/Users/Windows User/Desktop/UB Spring 2019/Computer Vision/Assignment 2/CPEG540\_tutorials/CPEG-540\_tutorials/profilepic.png'

img = cv2.imread(filename)

gray = cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY)

# find Harris corners

gray = np.float32(gray)

dst = cv2.cornerHarris(gray,2,3,0.04)

#result is dilated for marking the corners, not important

dst = cv2.dilate(dst,None)

# Threshold for an optimal value, it may vary depending on the image.

img[dst>0.01\*dst.max()]=[0,0,255]

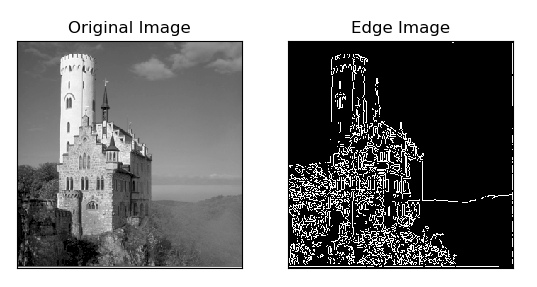
cv2.imshow('dst',img)

#-------------------------------------------------

cv2.waitKey(0)

cv2.destroyAllWindows()

First applying the Canny Edge Detection:



Further applying Harris Corner detection:



Furthermore, in C#, a Windows Form was created to allow us to load an image, and test both algorithms and see the results on the image. Through the Emgu CV library we are able to use the algorithms in the same fashion as with Python, as one just has to do a bit of research on how exactly to call the function and what parameters it is expecting and what should those be depending on what we need. For example, we still need to pass it to Grayscale to be able to call it.

In C#, we also needed to import a matrix class to allow us to get an image or possibly a bitmap from another program and convert its contents into a matrix so that it can be passed to both algorithms using the Emgu CV library.

Below is the source code for the C# project that calls both algorithms as well as the results visualized in the image.