CENG 391 Introduction to Image Understanding

November 3, 2016

Harris Corner Detector

Write a C++/Python program that takes an argument as the name of an image and operates the following tasks.

1. Read the image (”img1.ppm”)

2. Calculate first gradient images in both x and y directions and obtain I_x and I_y.

3. Then define a Gaussian window as follows: $G_{x,y} = \exp\frac{-(x^2+y^2)}{2\sigma^2}$ where x and y are the coordinates of the corresponding pixel.

4. Obtain following images:
   — I_A = G(I_x^2)
   — I_B = G(I_y^2)
   — I_C = G($I_x I_y$) where G denotes the smoothing window.

5. Construct a matrix S so that: $\begin{bmatrix} I_A & I_C \\ I_C & I_B \end{bmatrix}$

6. Then calculate corner response of the pixel p by the following formula: $R_p = \lambda(S)_p - k * T(S)_p^2$, where

   — $\lambda(S)$ is the determinant of the matrix S and can be found as follows: $\lambda(S) = I_A I_B - I_C^2$.

— $T(S)$ is the trace of the matrix S and can be found as follows: $T(S) = I_A + I_B$.

Value of the R_p determines the type of the region that pixel p locates as it is indicated in the below.
— R_p is a large for corner.
— R_p is negative with large magnitude for an edge.
— $|R_p|$ is small for a flat region.

7. Applying non-maxima supression. Pick 8-way local maximas as corner canditates.

8. You should determine the threshold for R on your own and construct a set of corners.

9. Draw final set of corners on the image by using "drawKeypoints()" method of OpenCV and save the image as "img1_corners.png".

NOTE: Take $k = 0.04$ and $\sigma = 2.0$