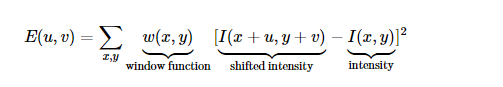
**Lab #02**

**Harris Corner Detection**

In the last chapter, we saw that corners are regions in the image with large variation in intensity in all the directions. One early attempt to find these corners was done by **Chris Harris & Mike Stephens** in their paper **A Combined Corner and Edge Detector** in 1988, so now it is called the Harris Corner Detector. He took this simple idea to a mathematical form. It basically finds the difference in intensity for a displacement of (u,v) in all directions. This is expressed as below:

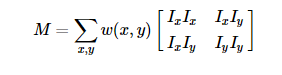


The window function is either a rectangular window or a Gaussian window which gives weights to pixels underneath.

We have to maximize this function E(u,v) for corner detection. That means we have to maximize the second term. Applying Taylor Expansion to the above equation and using some mathematical steps we get the final equation as:



Where,



Here, Ix and Iy are image derivatives in x and y directions respectively. (These can be easily found using [**cv.Sobel()**](https://docs.opencv.org/4.x/d4/d86/group__imgproc__filter.html#gacea54f142e81b6758cb6f375ce782c8d)).

Then comes the main part. After this, they created a score, basically an equation, which determines if a window can contain a corner or not.

R=det(M)−k(trace(M))2

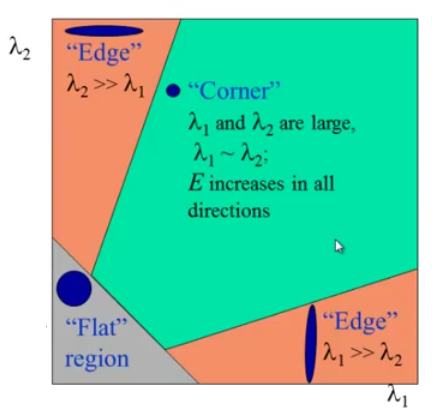
where

* det(M)=λ1λ2
* trace(M)=λ1+λ2
* λ1 and λ2 are the eigenvalues of M

So the magnitudes of these eigenvalues decide whether a region is a corner, an edge, or flat.

* When |R| is small, which happens when λ1 and λ2 are small, the region is flat.
* When R<0, which happens when λ1>>λ2 or vice versa, the region is edge.
* When R is large, which happens when λ1 and λ2 are large and λ1∼λ2, the region is a corner.

It can be represented as follows:



**image**

**TASK**:  Write a code for Harris Corner Point Detector by yourself. You can take help from the book "Learning OpenCV 3 Computer Vision with Python".