**目标：**

* 我了解Harris Corner Detection背后的概念。
* 函数：cv.cornerHarris()，cv.cornerSubPix()

## 理论

在上一节我们已经知道了角点的一个特性：向任何方向移动变化都很大。Chris\_Harris 和 Mike\_Stephens 早在 1988 年的文章《A CombinedCorner and Edge Detector》中就已经提出了焦点检测的方法，被称为Harris 角点检测。他把这个简单的想法转换成了数学形式。将窗口向各个方向移动（u，v）然后计算所有差异的总和

窗口函数可以是正常的矩形窗口也可以是对每一个像素给予不同权重的高斯窗口。

角点检测中要使 E (µ,ν) 的值最大。这就是说必须使方程右侧的第二项的取值最大。对上面的等式进行泰勒级数展开然后再通过几步数学换算（可以参考其他标准教材），我们得到下面的等式：

𝐸(𝑢,𝑣)≈[𝑢𝑣]𝑀[𝑢 𝑣]

其中：

𝑀=∑𝑥,𝑦𝑤(𝑥,𝑦)[𝐼𝑥𝐼𝑥𝐼𝑥𝐼𝑦 𝐼𝑥𝐼𝑦𝐼𝑦𝐼𝑦]

这里，$$I\_x$$和$$I\_y$$分别是x和y方向上的图像导数。（可以使用函数cv.Sobel()轻松找到）。

然后是主要部分。在此之后，他们创建了一个分数，基本上是一个等式，它将确定一个窗口是否可以包含一个角点。

𝑅=𝑑𝑒𝑡(𝑀)−𝑘(𝑡𝑟𝑎𝑐𝑒(𝑀))2

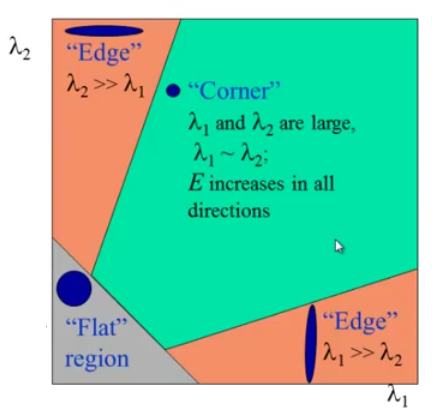
其中：

* 𝑅=𝑑𝑒𝑡(𝑀)−𝑘(𝑡𝑟𝑎𝑐𝑒(𝑀))2
* 𝑡𝑟𝑎𝑐𝑒(𝑀)=𝜆1+𝜆2
* 𝜆1和$$\lambda\_2$$是M的本征值

因此，这些特征值的值决定区域是角点，边缘还是平坦。

* 当$|R|$很小，当$\lambda\_1$和$\lambda\_2$很小时，该区域是平坦的。
* 当$R<0$时，在$\lambda\_1 >> \lambda\_2$时发生，反之亦然，该区域是边缘。
* 当R很大时，当$\lambda\_1$和$\lambda\_2$大并且$\lambda\_1 \sim \lambda\_2$时发生，该区域是拐角。

它可以用下面这张很好理解的图片表示：

[](https://raw.githubusercontent.com/TonyStark1997/OpenCV-Python/master/5.Feature%20Detection%20and%20Description/Image/image3.jpg)

因此Harris角点检测的结果是一个由角点分数构成的灰度图像。选取适当的阈值对结果图像进行二值化我们就检测到了图像中的角点。我们将用一个简单的图片来演示一下。

### 示例1

|  |
| --- |
|  |

### 效果

|  |
| --- |
|  |

## 具有亚像素精度的角点

有时，你可能需要以最高精度找到角点。OpenCV附带了一个函数cv.cornerSubPix()，它进一步细化了以亚像素精度检测到的角点。以下是一个例子。像往常一样，我们需要先找到Harris的角点。然后我们传递这些角的质心（角点处可能有一堆像素，我们采用它们的质心）来细化它们。Harris角以红色像素标记，精致角以绿色像素标记。对于此函数，我们必须定义何时停止迭代的标准。我们在指定的迭代次数或达到一定精度后停止它，以先发生者为准。我们还需要定义它将搜索角点的邻域大小。

### 示例2.

|  |  |
| --- | --- |
|  | **import cv2 import numpy as np  img = cv2.imread('../pic\_datas/chessboard-2.png') *# 图片太大，需要缩小 # img = cv2.resize(img,(600,480))* gray = cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY) gray = np.float32(gray) *#角点检测* dst = cv2.cornerHarris(gray,2,3,0.04) *#result is dilated for marking the corners, not important* dst = cv2.dilate(dst,None) *# 二值化* ret, dst = cv2.threshold(dst, 0.01 \* dst.max(), 255, 0) dst = np.uint8(dst)  *# find centroids # connectedComponentsWithStats(InputArray image, OutputArray labels, OutputArray stats, # OutputArray centroids, int connectivity=8, int ltype=CV\_32S)* ret,labels,stat,centerids = cv2.connectedComponentsWithStats(dst)  criteria = (cv2.TERM\_CRITERIA\_EPS + cv2.TERM\_CRITERIA\_MAX\_ITER,100,0.001) *# Python: cv2.cornerSubPix(image, corners, winSize, zeroZone, criteria) # zeroZone – Half of the size of the dead region in the middle of the search zone # over which the summation in the formula below is not done. It is used sometimes # to avoid possible singularities of the autocorrelation matrix. The value of (-1,-1) # indicates that there is no such a size. # 返回值由 点坐标组成的一个数组* corners = cv2.cornerSubPix(gray,np.float32(centerids),(3,3),(-1,-1),criteria)  *# Now draw them* res = np.hstack((centerids,corners)) *# np.int0 可以用来省略小数点后的数字，非四舍五入* res = np.int0(res) img[res[:, 1], res[:, 0]] = [0, 0, 255] img[res[:, 3], res[:, 2]] = [0, 255, 0]  cv2.imshow('subpixel5.png', img) cv2.waitKey(0) cv2.destroyAllWindows()** |

### 效果：

|  |
| --- |
|  |