**目标：**

* SURF的基础知识
* 在OpenCV中看到SURF功能

## api：

### surf = cv2.xfeatures2d\_SURF.create(400)

### surf.setHessianThreshold(50000) *#设置hessian阈值*

### surf.setUpright(True) # 使得所有关键点都指向上面

### surf.setExtended(True) # 设置描述符为128位

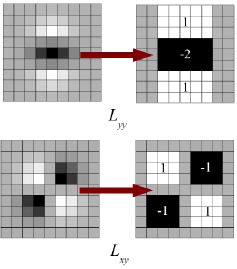
### kp,des = surf.detectAndCompute(img,None)

### kp = surf.detect(img,None)

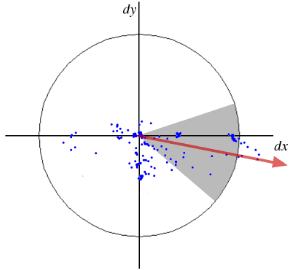
## 理论

在上一节中我们学习了使用 SIFT 算法进行关键点检测和描述。但是这种算法的执行速度比较慢，人们需要速度更快的算法。在 2006 年Bay,H.,Tuytelaars,T. 和 Van Gool,L 共同提出了 SURF（加速稳健特征）算法。跟它的名字一样，这是个算法是加速版的 SIFT。

在 SIFT 中，Lowe 在构建尺度空间时使用 DoG 对 LoG 进行近似。SURF则更进一步，使用盒子滤波器（box\_filter）对 LoG 进行近似。下图显示了这种近似。在进行卷积计算时可以利用积分图 像（积分图像的一大特点是：计算图像中某个窗口内所有像素和时，计算量的大小与窗口大小无关），是盒子滤波器的一大优点。而且这种计算可以在不同尺度空间同时进行。同样 SURF 算法计算关键点的尺度和位置是也是依赖与 Hessian 矩阵行列式的。

[](https://camo.githubusercontent.com/05fe28020ba5ccc61e8cb4683befd9f569bac3db83392d65bfba266f854f816c/68747470733a2f2f646f63732e6f70656e63762e6f72672f342e302e302f737572665f626f7866696c7465722e6a7067)

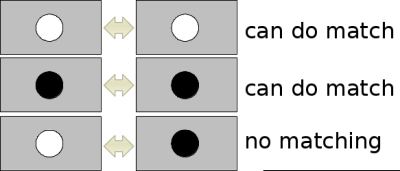
为了保证特征矢量具有选装不变形，需要对于每一个特征点分配一个主要方向。需要以特征点为中心，以 6s（s 为特征点的尺度）为半径的圆形区域内，对图像进行 Harr 小波相应运算。这样做实际就是对图像进行梯度运算，但是利用积分图像，可以提高计算图像梯度的效率，为了求取主方向值，需哟啊设计一个以方向为中心，张角为 60 度的扇形滑动窗口，以步长为 0.2 弧度左右旋转这个滑动窗口，并对窗口内的图像 Haar 小波的响应值进行累加。主方向为最大的 Haar 响应累加值对应的方向。在很多应用中根本就不需要旋转不变性，所以没有必要确定它们的方向，如果不计算方向的话，又可以使算法提速。SURF 提供了成为 U-SURF 的功能，它具有更快的速度，同时保持了对$\pm 15^{\circ}$旋转的稳定性。OpenCV 对这两种模式同样支持，只需要对参数upright 进行设置，当 upright 为 0 时计算方向，为 1 时不计算方向，同时速度更快。

[](https://camo.githubusercontent.com/60cc8be113727a5a5138a99cc254472f148f87b3c474ab40815da7817d6b7c64/68747470733a2f2f646f63732e6f70656e63762e6f72672f342e302e302f737572665f6f7269656e746174696f6e2e6a7067)

生成特征点的特征矢量需要计算图像的 Haar 小波响应。在一个矩形的区域内，以特征点为中心，沿主方向将 20s \* 20s 的图像划分成 4 \* 4 个子块，每个子块利用尺寸 2s 的 Haar 小波模版进行响应计算，然后对响应值进行统计，组成向量$v=( \sum{d\_x}, \sum{d\_y}, \sum{|d\_x|}, \sum{|d\_y|})$。这个描述符的长度为 64。降低的维度可以加速计算和匹配，但又能提供更容易区分的特征。

为了增加特征点的独特性，SURF 还提供了一个加强版 128 维的特征描述符。当$d\_y>0$和$d\_y<0$时分别对$d\_x$和$|d\_x|$的和进行计算，计算$d\_y$和$|d\_x|$时也进行区分，这样获得特征就会加倍，但又不会增加计算的复杂度。OpenCV 同样提供了这种功能，当参数 extended 设置为 1 时为 128 维，当参数为 0 时为 64 维，默认情况为 128 维。

另一个重要的改进是使用拉普拉斯算子（Hessian矩阵的迹线）作为潜在兴趣点。它不会增加计算成本，因为它已经在检测期间计算出来。拉普拉斯的标志将黑暗背景上的明亮斑点与相反情况区分开来。在匹配阶段，我们只比较具有相同类型对比度的特征（如下图所示）。这种最小的信息允许更快的匹配，而不会降低描述符的性能。

[](https://camo.githubusercontent.com/52d32556d9e0c6e1cb0321590e500f0073970eba27258f90e9ca4c7b7bc772c5/68747470733a2f2f646f63732e6f70656e63762e6f72672f342e302e302f737572665f6d61746368696e672e6a7067)

简单来说 SURF 算法采用了很多方法来对每一步进行优化从而提高速度。分析显示在结果效果相当的情况下 SURF 的速度是 SIFT 的 3 倍。SURF 善于处理具有模糊和旋转的图像，但是不善于处理视角变化和关照变化。

## OpenCV中的SURF

OpenCV就像SIFT一样提供SURF功能。首先使用一些可选条件（如64/128-dim描述符，Upright/Normal SURF等）初始化一个SURF对象。所有详细信息都在文档中进行了详细说明。然后就像我们在SIFT中所做的那样，我们可以使用SURF.detect()，SURF.compute()等来查找关键点和描述符。

## 实例1.

|  |
| --- |
|  |

### 效果：

|  |
| --- |
|  |

## 实例2.

|  |
| --- |
|  |

### 效果

|  |
| --- |
|  |

## 实例3.

|  |
| --- |
|  |

### 效果

|  |
| --- |
|  |