# 程序说明

**算法流程：**

线段提取算法LSD

SIFT描述子

欧式距离

暴力匹配算法

欧式距离

暴力匹配算法

恢复R，t算法

筛选机制

# 基础知识：

算法参考：opencv sift描述子算法

PL\_Slam LSD线段检测算法

Orb\_Slam 单目初始化时用到的R，t解算算法

LSD线段提取算法：

输入：灰度图像

输出：检测到的线段端点

为了实现尺度不变性，先对原图像做金字塔，每层金字塔上检测线段，剔除很短的线段；

将线段端点设为关键点，并在同一层金字塔上将靠的很近的端点选择其一，剔除另一个。

设想：后来跑Slam的过程中帧间相似度，缩放尺度很相近，是否有必要使用金字塔；另外需要在金字塔上提取关键点，还是只保留原始图像的关键点，而只需在金字塔上计算其描述子；靠的很近的关键点是剔除其一还是求其中点比较好。

LSD线段提取算法原理见链接：

<https://blog.csdn.net/tianwaifeimao/article/details/17678669>

<https://www.cnblogs.com/Jessica-jie/p/7510931.html>

论文：

<http://www.ipol.im/pub/art/2012/gjmr-lsd/article.pdf>

《基于图像分割的线段检测方法》 比较分析了几种线段检测算法

推荐学习网站，里面可以演示运行结果：

<http://www.ipol.im/pub/art/2012/gjmr-lsd/>

<https://zhuanlan.zhihu.com/p/54126417>

高斯金字塔：

输入：图像

输出：n个缩小的图像

相当于对图像进行缩小，只不过在缩小之前先做一次高斯滤波

<https://blog.csdn.net/poem_qianmo/article/details/26157633?tdsourcetag=s_pctim_aiomsg>

SIFT描述子：

输入：图像及关键点

输出：关键点对应的128位描述子

<https://www.cnblogs.com/JiePro/p/sift_4.html>

基于欧氏距离的暴力匹配：

首先在第一幅图像中选取一个关键点然后依次与第二幅图像的每个关键点进行(描述符)距离测试,最后返回距离最近的关键点。

欧氏距离

<https://blog.csdn.net/GAN_player/article/details/78285771>

设想：1，改用角度作为判断相似性；

2，换用别的匹配方法，比如FLANN特征匹配，设置布尔变量 crossCheck,默认值为 False。如果设置为True,匹配条件就会更加严格,只有到 A 中的第 i 个特征点与 B 中的第 j 个特征点距离最近,并且 B 中的第 j 个特征点到 A 中的第 i 个特征点也是最近(A 中没有其他点到 j 的距离更近)时才会返回最佳匹配(i,j)。

关键点去畸变：

<https://blog.csdn.net/weixin_39752599/article/details/82389555>

计算基础矩阵与单应矩阵：

<https://blog.csdn.net/Hansry/article/details/75000765>

<https://blog.csdn.net/luohuiwu/article/details/80722439>

从基础矩阵恢复R,t:

<https://www.cnblogs.com/houkai/p/6665506.html>

<https://blog.csdn.net/scyscyao/article/details/6220221>

<https://blog.csdn.net/kokerf/article/details/72885435>

论文：

<https://www.researchgate.net/publication/243764888_Motion_and_Structure_from_Motion_in_a_Piecewise_Planar_Environment>

**源码解读：**

1. 提取特征点，计算描述子：

sle\_frame->detect\_keypoints( gray\_, keypoints\_ , descriptors\_ );

gray\_ 输入的灰度图像；

keypoints\_ 存储关键点的的容器变量

descriptors\_ 存储关键点对应的描述子的Mat变量

void Sift\_Line\_Endpoint::detect\_keypoints(const Mat &image, std::vector<KeyPoint> &Kpoints, Mat& desc\_){

img = image.clone();

buildGaussianPyramid();

LSDDetectorC::LSDOptions options;

options.refine = LSD\_REFINE\_ADV;

options.scale = 0.8;

options.sigma\_scale = 0.6;

options.quant = 2.0;

options.ang\_th = 22.5;

options.log\_eps = 1.0;

options.density\_th = 0.6;

options.n\_bins = 1024;

options.min\_length = 10;

get\_lsdoptions(options);

get\_gpyrkeypoints();

Kpoints.assign(keypoints.begin(),keypoints.end())

calc\_Gpyr\_Orientation\_Hist(5,2,36);

calculator\_gpyr\_descriptor();

desc\_=descriptors.clone();

keypoints.clear();

}

建立图像金字塔（没有用高斯模糊，仅进行下采样）；

Options是LSD线段检测算法的相关参数；

在各个金字塔图像上计算关键点，剔除关键点；

计算关键点的主方向

在关键点对应的金字塔图像上计算SIFT描述子

calc\_Gpyr\_Orientation\_Hist(5,2,36);

计算关键点的主方向

5----计算范围的半径为5；

2----高斯权重系数

36---计算的时候将一圆周均分为36等分

calcSIFTDescriptor(des\_img, ptf, angle, 3, d, n, descriptors.ptr<float>((int)i));

计算描述子

des\_img ： 图像

ptf： 关键点

angle： 关键点对应的主方向

3 ----决定计算半径的一个参数；

d = 4 ---- SIFT\_DESCR\_WIDTH 描述子直方图的宽度，也用来计算计算半径

n = 8 ---- 描述子向量每个直方图的bin数，也就是将360度分为8个方向

描述子的长度为4\*4\*8；

设问：参数选择的原理，目前都是参考opencv库函数的选择方法

计算方向的时候选的半径是否需要和后来计算描述子的半径一样

暴力匹配：

matcher.match(ref\_->descriptors\_,curr\_->descriptors\_ , matches);

输入：两个描述子集合；

输出：配对

设想：1，改用角度作为判断相似性；

2，换用别的匹配方法，比如FLANN特征匹配，设置布尔变量 crossCheck,默认值为 False。如果设置为True,匹配条件就会更加严格,只有到 A 中的第 i 个特征点与 B 中的第 j 个特征点距离最近,并且 B 中的第 j 个特征点到 A 中的第 i 个特征点也是最近(A 中没有其他点到 j 的距离更近)时才会返回最佳匹配(i,j)。

关键点去畸变：

cv::undistortPoints(mat,mat,mK,mDistCoef,cv::Mat(),mK);

输入： 关键点Mat数据集合，相机内参，相机畸变参数

输出： 去畸变后的点集合

计算单应矩阵和基础矩阵：

FindHomography(vbMatchesInliersH, SH, H);

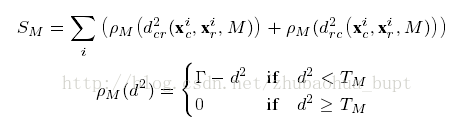
FindFundamental(vbMatchesInliersF, SF, F);

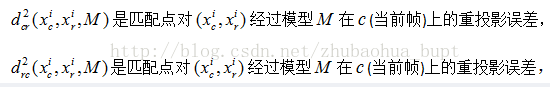
输入：

RANSANC挑选出来的八对匹配点：vbMatchesInliersH，vbMatchesInliersF

输出： 单应矩阵H和基础矩阵F，SH,SF得分。

SH,SF计算方法：先有八对点计算得到H,F，然后对所有的匹配点根据下列公式计算SH,SF

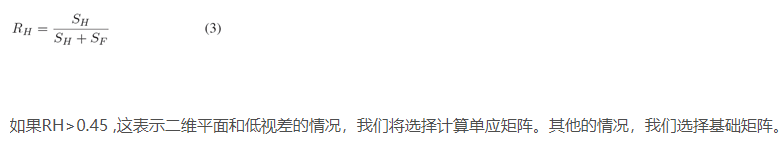




TM是无效数据的排除阈值，

它的依据是卡方测试的95%（TH=5.99, TF=3.84，假设在测量误差上有1个像素的标准偏差）。 TM等于TH，这样，对于同一个重投影误差d，两个模型求取的分值相同。模型M越准，所有匹配点对的重投影误差越小，SM的值就越大。因此SM的值反映了一个这样的问题，当其越大时，模型M越准确。

选择H或者F矩阵：



从H或者F恢复得到R,t：

首选从H,F计算得到几个R，t（H可以得到8个，F可以得到4个），然后用三角化得到特征点在空间中的深度，对于某一个点而言，计算结果有效需符合以下几个条件：

1. 特征点在两帧坐标系下的深度都为正；
2. 特征点投影到两个图像坐标系下的重投影误差小于一个阈值（设为4）；

不符合上述条件则代表此R,t是无效的，或者说这个特征点对应该R,t是无效的，统计所有的R,t组合里有效特征点最多的即为所求R，t； 如果有效点最多的一组R，t得到的有效点数量达不到所有点（100）的一半，或者有多组R，t对应的有效点数都多于最大有效点数的70%（此代表没有明显有优势的一组解），则此次不返回运算结果，视为无效。为了排除掉相机不动的情况，如果算出来的R,t 得到的特征点空间坐标对应于两个光心的夹角小于一个阈值（orb\_slam里设为1度），也不返回计算结果；

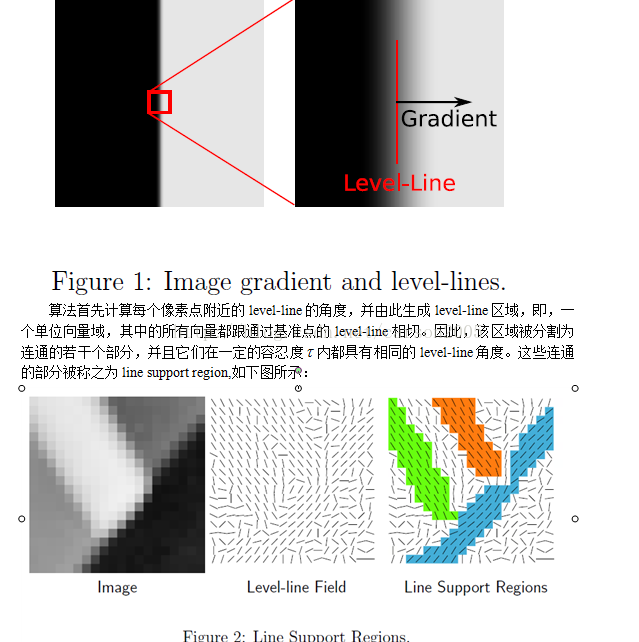
## LSD直线检测算法

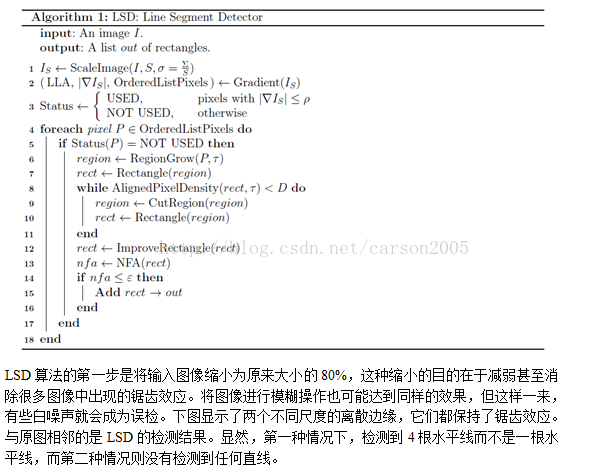
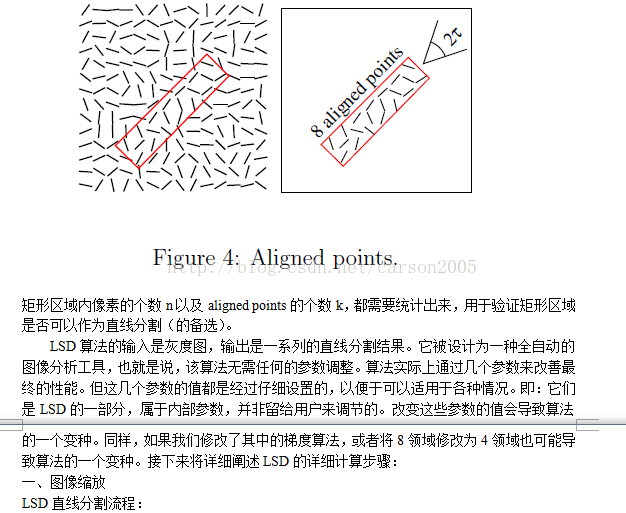
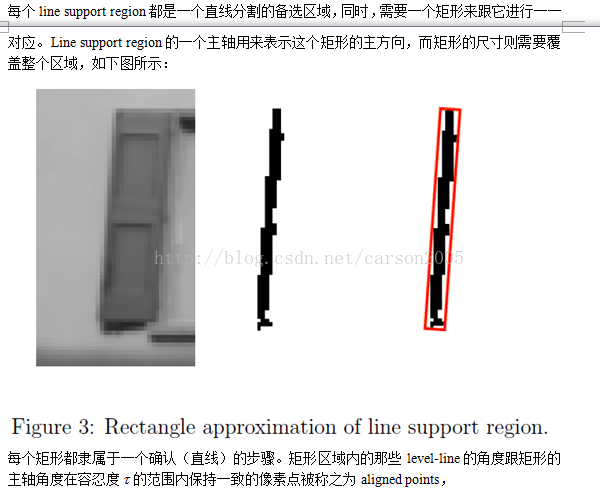
LSD是一种直线检测分割算法，它能在线性的时间内得出亚像素级精度的检测结果。

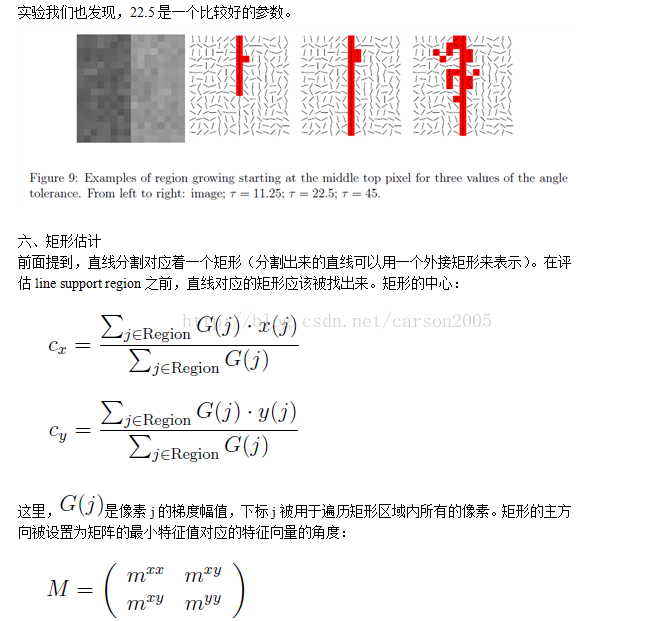
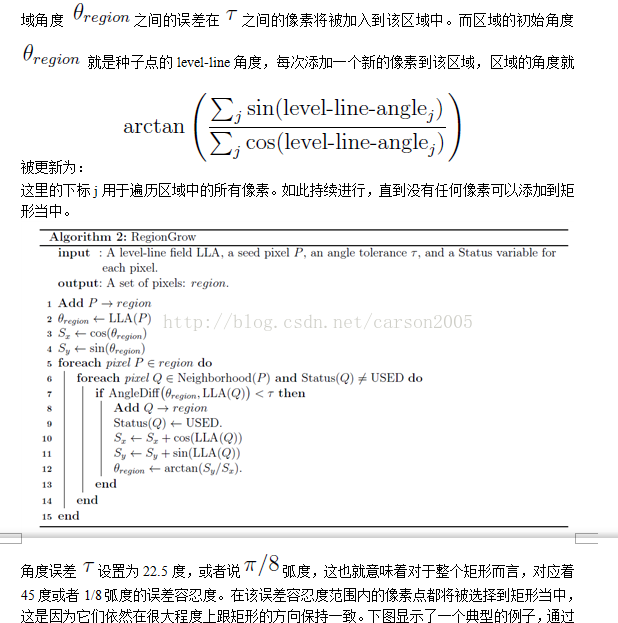
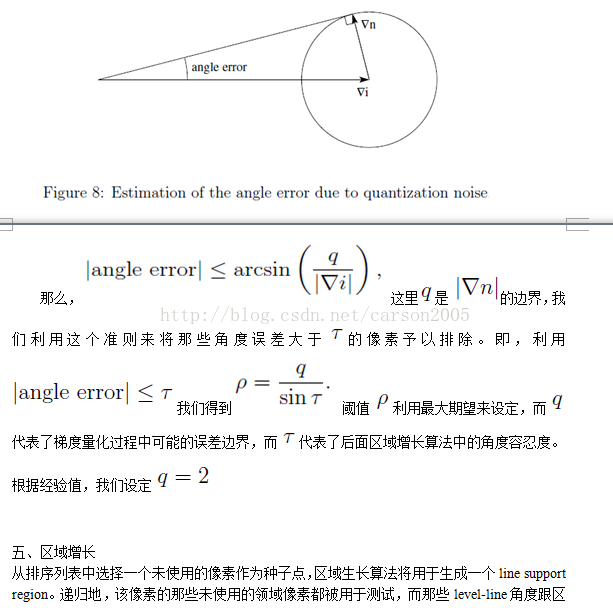
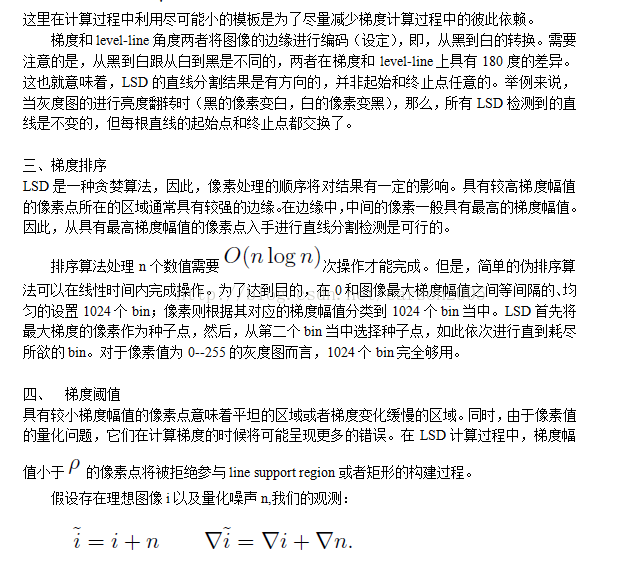
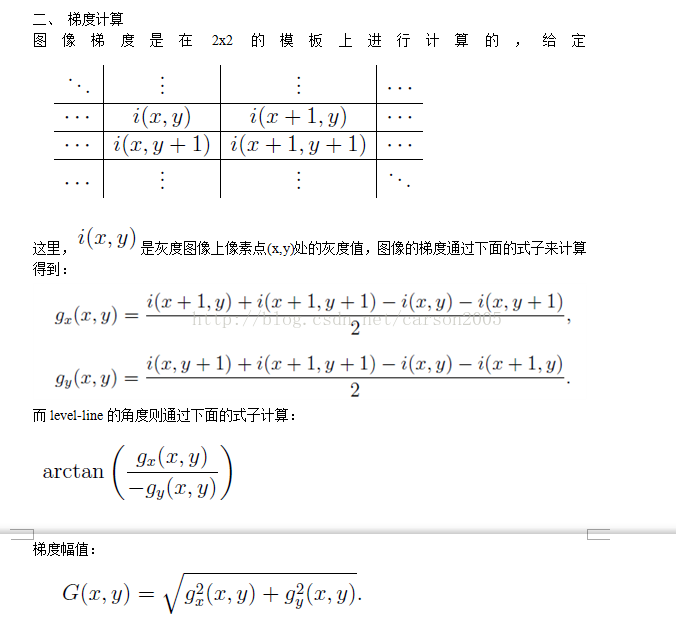
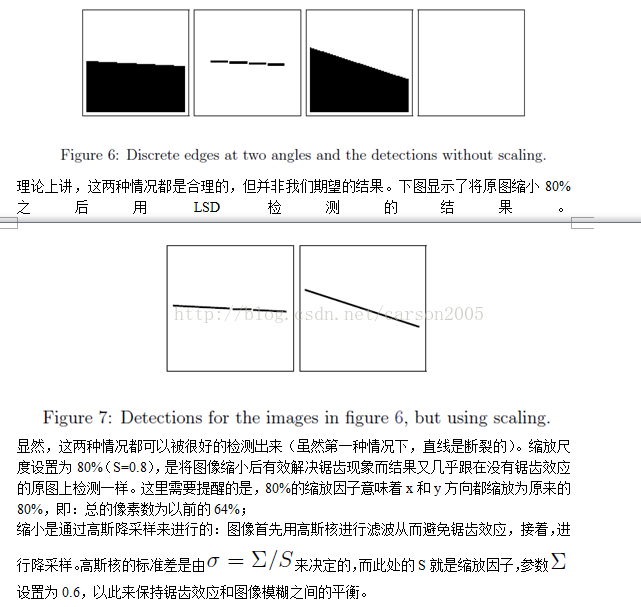
https://www.cnblogs.com/Jessica-jie/p/7512152.html

  LSD是一种直线检测分割算法，它能在线性的时间内得出亚像素级精度的检测结果。该算法被设计成可以在任何数字图像上都无需参数调节。它可以自己控制误检的数量：平均而言，每张图有一个误检。

LSD的目标在于检测图像中局部的直的轮廓。这也是我们称之为直线分割的原因。轮廓是图像中的某些特殊区域，在这些区域，图像的灰度从黑到白或者从白到黑的剧烈变化。因此，梯度和level-line是两个重要的概念，如下图所示：







【7】NFA计算

**自己的理解：**其实这个NFA，就是通过你的观察图中矩形中的aligned points所占比例的大小与引入的一个新的模型 （a contrario model）的aligned points作比较的概率，来约束这个矩形是不是可以作为一条“线段”。（就如同作者所希望的，这个aligned points越多，越有可能是一条“线段”）

**矩形验证：**

用来验证矩形是否可以作为检测线段的方法是基于a contrario approach and the Helmholtz principle .

the Helmholtz principle：在完美噪声图像图像中不应该检测到目标。

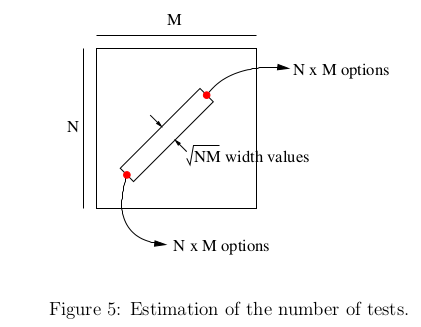
a  contrario approach：一个不会检测到目标的噪声图像。

对于本课题，contrario model  H0是一个像素值为level-line angle的图像，其level-line angle随机分解为独立且服从平均分布于[0,2π]。

（即主要有以下两个属性： （1）{LLA(j)}，是 level-line场，其中j是像素，由独立随机变量组成；（2）LLA(j)在[0,2π]上均匀分布。 ）

这里用NFA（Number of False Alarms）来评判observe img中某个候选R小于contrario model中相同位置R里同性点（aligned  points）的数量的概率，如果NFA的值很大，认为在观测图像中aligned points比contrario model中aligned points小的概率很大，将其认为是common，平常的，背景中的一部分。如果NFA的值很小，认为目标是相对突出（rare）的，是一个合适的“直线”。

https://images2017.cnblogs.com/blog/1034872/201709/1034872-20170914090051735-904366074.png



其中，Ntest为当前大小（n\*m）图像中直线（矩形框）的数量。在n\*m的图像中直线的起点和终点分别有m\*n种选择，所以一共有(n\*m）\*(n\*m)种起点和终点搭配。线段的宽度为(n\*m)^0.5，因此在m\*n大小的图像中有（n\*m）^2.5 种不同直线。The precision p is initially set to the value τ /π.We will note γ the number different p values potentially tried. Each rectangle with each p value is a different test.Thus, the final number of tests ishttps://images2017.cnblogs.com/blog/1034872/201709/1034872-20170914135046297-1402636477.png

PH0是对应 contrario model H0 的一个概率，I是在模型H0下的随机图像。（H0是图像梯度方向的噪声模型而不是一个图像的噪声模型。）

k(r,I) 为contrario model ，I 中 r 矩形里aligned pt的数量。

k(r,i) 为observe img，i 中 r 矩形里aligned pt的数量。

一个关键概念是p-aligned points，是指矩形中的level-line angle与最小外接矩形的主方向的角度差在容忍（tolerance）pπ内的像素点。 precision p最初设置的值为τ /π（角度正负容忍误差为τ ，总容忍误差为2τ 。那么在contrario model中，某个点为aligned point的概率为 p=2\*τ / 2\*π =τ / π），

k(r, I) 服从二项分布，所以：

https://images2017.cnblogs.com/blog/1034872/201709/1034872-20170914151727922-10262499.png

所以，NFA的数量与矩形r有关：

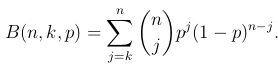
https://images2017.cnblogs.com/blog/1034872/201709/1034872-20170914134343985-740528326.png

简化r和i的公式：

https://images2017.cnblogs.com/blog/1034872/201709/1034872-20170914101013750-1382740193.png

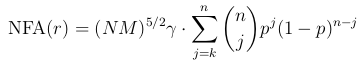
其中，N和M是采样过后图像的列和行，n：矩形的像素的总数，k：p-aligned点数。

二项分布：

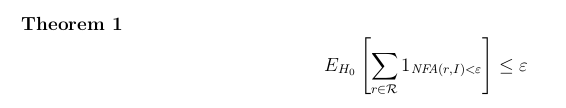


**解释：**设k(r,i)=k。那么，在 I 中的 r 矩形里，总像素个数为 n，I 中的 r 矩形里aligned pt个数k(r,I)大于等于k的话，可选择的值为k(r,i)、k(r,i)+1、k(r,i)+2，......n。

NFA计算：



**若NFA(r) ≤ ε，那么可以认为结果有效。**As stated before,we set **ε = 1** .

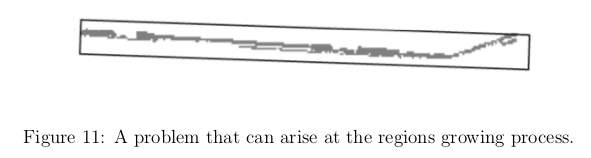


其中，E是期望算子，1是指示符函数，r是考虑的矩形集合，而I是H0上的随机图像。

 该定理指出在 contrario model H0上的ε-meaningful 矩形的平均数量小于ε。因此，噪声检测的次数通过ε控制的并且可以根据需要往小的调整。  
换句话说，这表明LSD满足亥姆霍兹原理（具体证明可以看原文）。

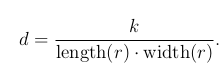
【8】Aligned Points Density

在某些情况下，这τ角度容忍度的方法产生一个错误的解释。这个问题可能会出现两直线边缘存在于图像形成一个夹角小于容忍度τ。如下图显示了一个线支持区域（灰色）和与其对应的矩形的例子：



在LSD中，这个问题是通过检测有问题的线支持区域并将其切割成两个较小的区域来处理的，希望在适当的位置将该区域分割出来以解决问题。这个“角度问题”的检测是基于矩形中同性点的密度。  
当这个问题不存在时，矩形非常适合于线支持区域，并且同性点的密度很高。另一方面，当“角问题”出现时，正如前面的图所看到的，同性点的密度很低。同样，当一个稍微弯曲的边被一系列直边近似地逼近时，近似的程度（多少线段被用来覆盖曲线的一部分）与同性点的密度有关；因此，对齐点密度也与线段近似曲线的精度有关。

同性点密度的计算公式：



规定一个阈值D ，在这里设置D=0.7，文章认为这个数字既能保证同一个R中的同性点属性相近，也能保证R不会被过分的分割为小的矩形。

若d > D（同性点密度阈值），accepted。否则，需要将R截断。

R截断的方法有两种：“缩小角度误差阈值”与“缩小区域半径”的方法。在这两种方法中，区域中的一部分像素被保留，而另一些则重新标记为NOT USED，因此它们可以在将来的线支持区域中再次使用。

缩小角度容忍阈值：简单的将τ值缩小，再次从当前R的seed开始搜寻，看是否满足要求。

缩小区域半径：逐渐去除远离种子的像素，直到满足标准或区域太小而被拒绝为止。当线支持区域对应于一条曲线时，该方法效果最好，并且在满足密度准则之前需要减少区域，通常意味着对曲线有一定程度的近似。

【9】矩形的改进

如果当前的R仍旧不能满足NFA(要求：**NFA(r) ≤ ε**)，以下的方法将对其进行改进。考虑到在有些情况下，删除line-support region中的一个点会减少R的 length-1个点（想象为对角线）。对不满足NFA的R，采取以下策略：

1.减小p=p/2

2.短边减少一行

3.长边减少一行

4.长边减少另一行

5.减小p=p/2

直至满足NFA（**NFA(r) ≤ ε**）或 R过小被拒 或 p为原来的1/32。



## SIFT描述子

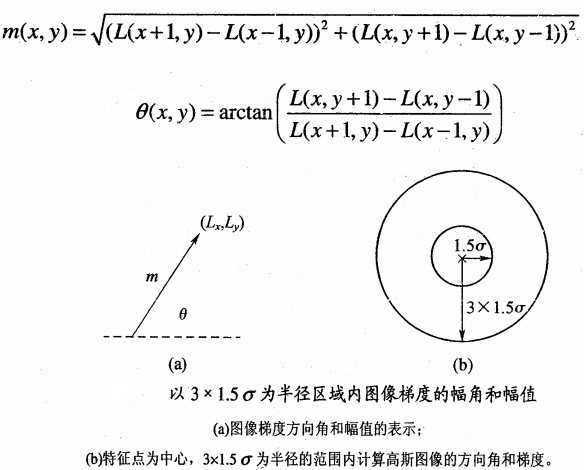
### 1，确定特征点方向

半径里的σ与原算法中特征点检测算法有关，在本算法中由于并没有这个量，因此设为10（原算法中一般的半径）

### https://www.cnblogs.com/JiePro/p/sift\_3.html

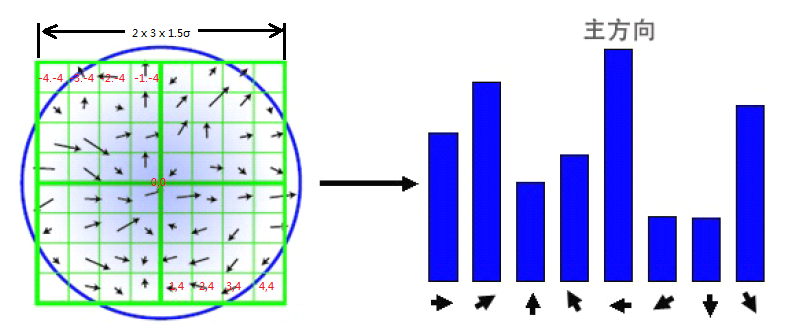
# 1 计算邻域梯度方向和幅值

　　为了实现图像旋转的不变性，需要根据检测到的特征点的局部图像结构求得一个方向基准。我们使用图像梯度的方法求取该局部结构的稳定方向。对于己经检测到特征点，我们知道该特征点的尺度值σ，因此根据这一尺度值，在GSS中得到最接近这一尺度值的高斯图像。然后使用有限差分，计算以特征点为中心，以3X1.5σ为半径的区域内图像梯度的幅角和幅值，如下图所示。幅角和幅值计算公式加下:



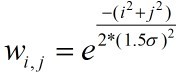
# 2 计算梯度方向直方图

　　在完成特征点邻域的高斯图像的梯度计算后，使用直方图统计邻域内像素的梯度方向和幅值。梯度方向直方图的横轴是梯度方向角，纵轴是梯度方向角对应的（带高斯权重）梯度幅值累加值。梯度方向直方图将。0°~360°的范围，分为36个柱，每10°为一个柱。直方图的峰值代表了该特征点处邻域内图像梯度的主方向，也即该特征点的主方向，如下图所示。



　　绿色格点代表邻域范围，蓝色圆圈代表格点的高斯权重（稍后介绍），黑色箭头指向代表梯度方向，箭头长度代表梯度幅值。右边为梯度方向直方图（36柱，每柱代表10°，上图只显示了8柱）。获得梯度方向直方图的步骤如下：

* 生成领域各像元的高斯权重。其中高斯函数方差为该特征点的特征尺度σ的1.5倍。形式如下，其中（i,j）为该点距离特征点的相对位置，如上图，左上角点像元距离特征点（0,0）（即中心点）的相对位置坐标为（-4,-4），同理，右下角像元为（4,4）。



* 遍历邻域（绿色）中每个点，判断其梯度方向，将其加入相应的梯度方向直方图中，加入量为其梯度幅值 \* wi,j，例如左上角(-4,-4)的点，其梯度为方向为25°，梯度幅值为mag，我们将其加入到hist[2]中（假设hist[0]为0°~10°的直方柱，hist[1]为10°~20°的直方柱，以此类推至hist[35]为350°~360°）。加入的量为mag\* w(-4,-4)，即hist[2] = hist[2] + mag\* w(-4,-4)。直至遍历整个邻域，统计出该特征点出的梯度方向直方图。
* 平滑直方图。对上一步得出的直方图进行平滑，得到最终的梯度方向直方图。OpenCV中使用的 (1/16) \* [1,4,6,4,1] 的高斯卷积和对直方图进行平滑处理，而vlfeat中使用了6次，邻域大小为3的平均处理，即hist[i] = (hist[i-1]+hist[i]+hist[i+1])/3。

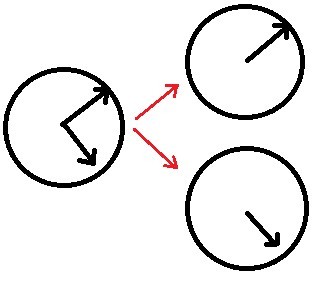
**问题1： 为什么每个点梯度幅值要使用高斯权重？**

答：由于SIFT算法只考虑了尺度和旋转的不变性，并没有考虑仿射不变性。通过对各点梯度幅值进行高斯加权，使特征点附近的梯度幅值有较大的权重，这样可以部分弥补因没有仿射不变性而产生的特征点不稳定的问题。

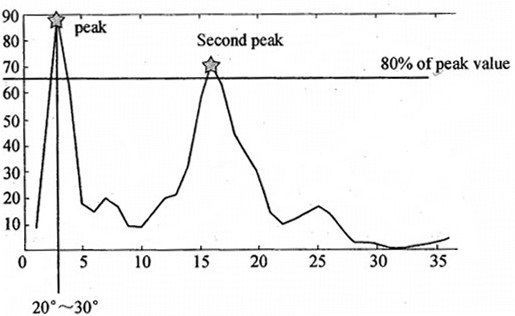
# 3 确定特征点方向

　　有了梯度方向直方图之后，找到直方图中最大的值，则认为该方向为该特征点的主方向，如存在另一个方向大于最大值的80％，则认为该方向为该特征点的辅方向。一个特征点可能会有多个方向（一个主方向，一个以上的辅方向），这可以增强匹配的鲁棒性。具体而言，就是将该特征点复制成多份特征点（除了方向θ不同外，x,y,σ都相同）。

【Note】在OpenCV中，若辅方向除了满足大于最大值80％外，还必须是局部最大值，即 hist[i] > hist[i-1] && hist[i] > hist[i+1]。



　　通常离散的梯度方向直方图，可以通过插值拟合处理，这样可以得到更精确的方向角度值。

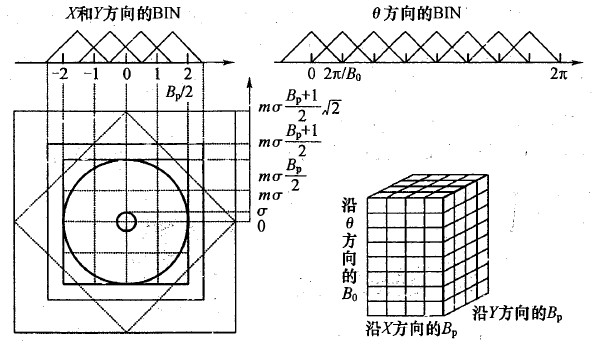


　　经过上述过程，我们特征点的所有量（x,y,σ,θ）都已经已经求得，其中位置（x,y）、尺度σ都是在[上一节](http://www.cnblogs.com/JiePro/p/sift_2.html)中求得，而特征点方向θ是通过特征点邻域直方图求得。下一节，将介绍SIFT描述子的形成方式。

### 2，特征描述子

# 1 确定描述子采样区域

　　SIFI 描述子h(x, y, θ)是对特征点附近邻域内高斯图像梯度统计结果的一种表示，它是一个三维的阵列，但通常将它表示成一个矢量。矢量是通过对三维阵列按一定规律进行排列得到的。特征描述子与特征点所在的尺度有关，因此，对梯度的求取应在特征点对应的高斯图像上进行。将特征点附近邻域划分成Bp X Bp个子区域，每个子区域的尺寸为mσ个像元，其中，m=3，Bp=4。σ为特征点的尺度值。考虑到实际计算时，需要采用双线性插值，计算的图像区域为mσ(Bp+ 1)。如果再考虑旋转的因素，那么，实际计算的图像区域应大mσ(Bp+ 1)√2。



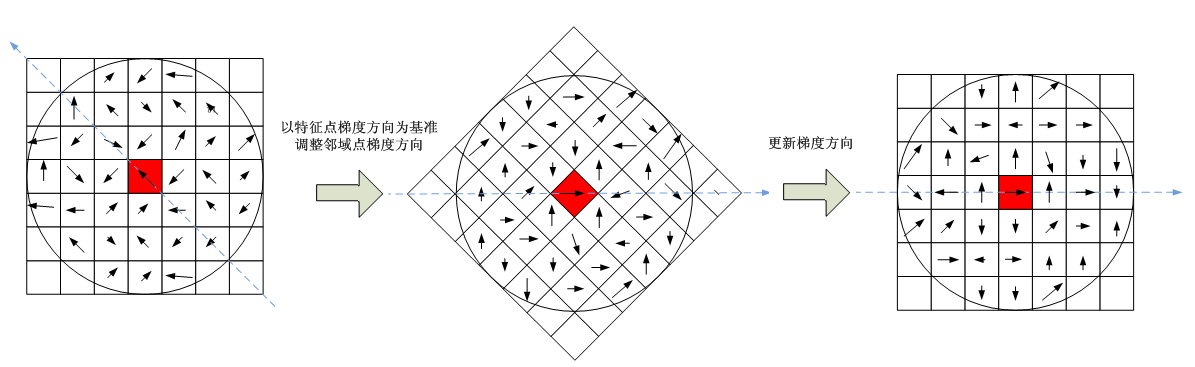
# 2 生成描述子

## 2.1 旋转图像至主方向

　　为了保证特征矢量具有旋转不变性，需要以特征点为中心，将特征点附近邻域内(mσ(Bp+ 1)√2 x mσ(Bp+ 1)√2)图像梯度的**位置**和**方向**旋转一个方向角θ，即将原图像x轴转到与主方向相同的方向。旋转公式如下。

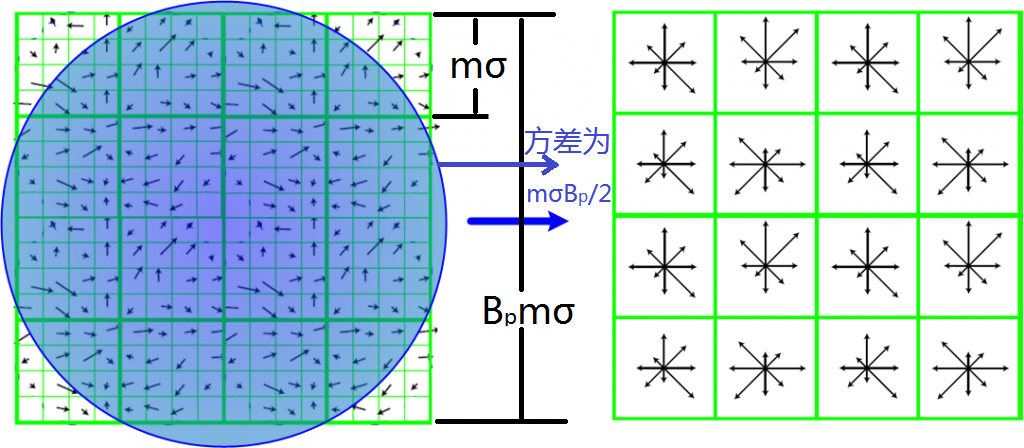
https://images0.cnblogs.com/blog/560725/201309/04173934-406501a77cd84918bcdc587d86b26437.jpg

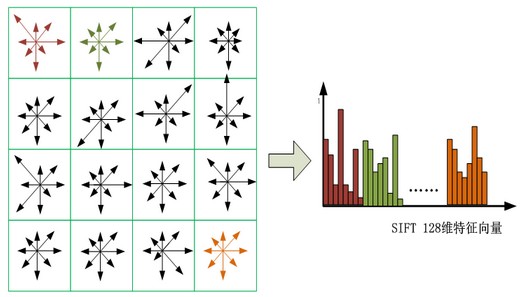
　　在特征点附近邻域图像梯度的位置和方向旋转后，**再以特征点为中心，在旋转后的图像中取一个mσBp x mσBp大小的图像区域。**并将它等间隔划分成Bp X Bp个子区域，每个间隔为mσ像元。



## 2.2 生成特征向量

　　在每子区域内计算8个方向的梯度方向直方图，绘制每个梯度方向的累加值，形成一个种子点。与求特征点主方向时有所不同，此时，每个子区域的梯度方向直方图将0° ~360°划分为8个方向范围，每个范围为45°，这样，每个种子点共有8个方向的梯度强度信息。由于存在4X4（Bp X Bp）个子区域，所以，共有4X4X8=128个数据，最终形成128维的SIFT特征矢量。同样，对于特征矢量需要进行高斯加权处理，加权采用方差为**mσBp/2**的标准高斯函数，其中距离为各点相对于特征点的距离。使用高斯权重的是为了防止位置微小的变化给特征向量带来很大的改变，并且给远离特征点的点赋予较小的权重，以防止错误的匹配。





# 3 归一化特征向量

　　为了去除光照变化的影响，需对上述生成的特征向量进行归一化处理，在归一化处理后，对特征矢量大于0.2的要进行截断处理，即大于0.2的值只取0.2，然后重新进行一次归一化处理，其目的是为了提高鉴别性。

## 目标定位

参考高博的《视觉SLAM十四讲》第七节：视觉里程计

