SIFT

第一章 SIFT 简介

1.1 SIFT 综述

尺度不变特征转换（Scale-invariant feature transform 或 SIFT）是一种电脑视觉的算法用来侦测与描述影像中的局部性特征，它在空间尺度中寻找极值点，并提取出其位置、尺度、旋转不变量，此算法由 David Lowe 在 1999 年所发表，2004 年完善总结。

其应用范围包含物体辨识、机器人地图感知与导航、影像缝合、3D 模型建立、手势辨识、影像追踪和动作比对。

此算法有其专利，专利拥有者为英属哥伦比亚大学。

SIFT 是局部不变特征，与 SIFT 类似的 SURF 也是局部不变特征。从直观的人类视觉印象来看，人类视觉对物体的描述也是局部化的，基于局部不变特征的图像识别方法十分接近于人类视觉机理，通过局部化的特征组合，形成对目标物体的整体印象，这就为局部不变特征提取方法提供了生物学上的解释，因此局部不变特征也得到了广泛应用。图像中的每个局部区域的重要性和影响范围并非同等重要，即特征不是同等显著的，其主要理论来源是 Marr 的计算机视觉理论和 Treisman 的特征整合理论，一般也称为“原子论”。该理论认为视觉的过程开始于对物体的特征性质和简单组成部分的分析，是从局部性质到大范围性质。SIFT/SURF 都是对特征点的局部区域的描述，这些特征点应该是影响重要的点，对这些点的分析更加重要。所以在局部不变特征的提取和描述时也遵循与人眼视觉注意选择原理相类似的机制，所以 SIFT/SURF 用于匹配有效果。

局部影像特征的描述与侦测可以帮助辨识物体，SIFT 特征是基于物体上的一些局部外观的兴趣点而与影像的大小和旋转无关。对于光线、噪声、些微视角改变的容忍度也相当高。基于这些特性，它们是高度显著而且相对容易撷取，在母数庞大的特征数据库中，很容易辨识物体而且鲜有误认。使用 SIFT 特征描述对于部分物体遮蔽的侦测率也相当高，甚至只需要 3 个以上的 SIFT 物体特征就足以计算出位置与方位。在现今的电脑硬件速度下和小型的特征数据库条件下，辨识速度可接近即时运算。SIFT 特征的信息量大，适合在海量数据库中快速准

确匹配。

SIFT 采用梯度作为局部不变特征，这与人的视觉神经相关。采用梯度作为描述子的原因是，人的视觉皮层上的神经元对特定方向和空间频率的梯度相应很敏感，经过 SIFT 作者的一些实验验证，用梯度的方法进行匹配效果很好。

SIFT 算法的特点有：

1. SIFT 特征是图像的局部特征，其对旋转、尺度缩放、亮度变化保持不变性，对视角变化、仿射变换、噪声也保持一定程度的稳定性；

2. 独特性（Distinctiveness）好，信息量丰富，适用于在海量特征数据库中进行快速、准确的匹配；

3. 多量性，即使少数的几个物体也可以产生大量的 SIFT 特征向量；

4. 高速性，经优化的 SIFT 匹配算法甚至可以达到实时的要求；

5. 可扩展性，可以很方便的与其他形式的特征向量进行联合。

SIFT 算法可以解决的问题：

目标的自身状态、场景所处的环境和成像器材的成像特性等因素影响图像配准/目标识别跟踪的性能。而 SIFT 算法在一定程度上可解决：

1. 目标的旋转、缩放、平移（RST）
2. 图像仿射/投影变换（视点 viewpoint）
3. 光照影响（illumination）
4. 目标遮挡（occlusion）
5. 杂物场景（clutter）
6. 噪声

SIFT 算法的实质是在不同的尺度空间上查找关键点(特征点)，并计算出关键点的方向。SIFT 所查找到的关键点是一些十分突出，不会因光照，仿射变换和噪音等因素而变化的点，如角点、边缘点、暗区的亮点及亮区的暗点等。

Lowe 将 SIFT 算法分解为如下四步：

1. 尺度空间极值检测：搜索所有尺度上的图像位置。通过高斯微分函数来识别潜在的对于尺度和旋转不变的兴趣点。

2. 关键点定位：在每个候选的位置上，通过一个拟合精细的模型来确定位置和尺度。关键点的选择依据于它们的稳定程度。

3. 方向确定：基于图像局部的梯度方向，分配给每个关键点位置一个或多个方向。所有后面的对图像数据的操作都相对于关键点的方向、尺度和位置进行变换，从而提供对于这些变换的不变性。

4. 关键点描述：在每个关键点周围的邻域内，在选定的尺度上测量图像局部的梯度。这些梯度被变换成一种表示，这种表示允许比较大的局部形状的变形和光照变化。

本文沿着 Lowe 的步骤，参考 Rob Hess 及 Andrea Vedaldi 源码，详解 SIFT 算法的实现过程。

SIFT 在图像的不变特征提取方面拥有无与伦比的优势，但并不完美，仍然存在：

1. 实时性不高。
2. 有时关键点较少。
3. 对边缘光滑的目标无法准确提取关键点。

等缺点，对模糊的图像和边缘平滑的图像，检测出的关键点过少，对圆更是无能为力。近来不断有人改进，其中最著名的有 ASIFT、SURF 和 CSIFT、PCA-SIFT 等。

相较于 SIFT 中的关键点提取方法，SIFT 中生成特征描述子的方法更加有名。你可以对通过其他方法获得的特征点使用 SIFT 描述子进行特征描述。以 Harris 为例，Harris 角点是在单一尺度下获取的关键点，而 SIFT 描述子是在多尺度下获得的，将它们结合在一起，一般来说有两种方法，一种是在多尺度上提取 Harris 角点，即在多尺度上的 Harris 角点检测，然后在多尺度上用 SIFT 中的方法生成特征描述子，然后匹配，这种当然是最好的。第二种是为了提高速度，直接在原始图像上提取角点，然后将 SIFT 中生成特征描述子的过程改到单尺度上进行，最后实现匹配。

第二章 尺度空间

2.1 尺度空间理论

自然界中的物体随着观测尺度不同有不同的表现形态。例如我们形容建筑物用“米”，观测分子、原子等用“纳米”。更形象的例子比如 Google 地图，滑动鼠

标轮可以改变观测地图的尺度，看到的地图绘制也不同；还有电影中的拉伸镜头等等……

尺度空间中各尺度图像的模糊程度逐渐变大，能够模拟人在距离目标由近到远时目标在视网膜上的形成过程。

尺度越大图像越模糊。

2.3 为什么讨论尺度空间

用机器视觉系统分析未知场景时，计算机并不预先知道图像中物体的尺度。我们需要同时考虑图像在多尺度下的描述，获知感兴趣物体的最佳尺度。另外如果不同的尺度下都有同样的关键点，那么在不同的尺度的输入图像下就都可以检测出来关键点匹配，也就是尺度不变性。

图像的尺度空间表达就是图像在所有尺度下的描述。

2.3 多分辨率金字塔

金字塔是早期图像多尺度的表示形式。图像金字塔化一般包括两个步骤：使用低通滤波器平滑图像；对平滑图像进行降采样（通常是水平，竖直方向 1/2），从而得到一系列**尺寸缩小**的图像。

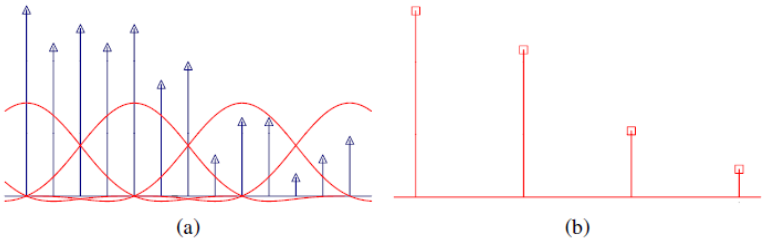


图 2-1 （a）是对原始信号进行低通滤波（b）是降采样得到的信号

而对于二维图像，一个传统的金字塔中，每一层图像由上一层分辨率的长、宽各一半，也就是四分之一的像素组成：

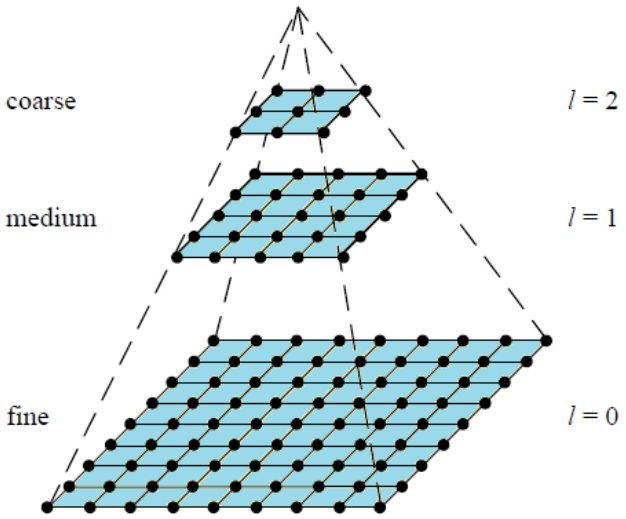


图 2-2 多分辨率金字塔

2.4 尺度空间表达

高斯核是唯一可以产生多尺度空间的核（《Scale-space theory: A basic tool for analysing structures at different scales》）。

一个图像的尺度空间 $L(x, y; \sigma)$ ，定义为原始图像 $I(x, y)$ 与一个可变尺度的二维高斯函数 $G(x, y; \sigma)$ 卷积运算。

一元高斯（正态分布）函数为：

$$G(x;\mu,\sigma)=\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{2.1}$$

多元高斯（正态分布）函数为：

$$G(x;\mu,\Sigma)=\frac{1}{(2\pi)^{\frac{N}{2}}}\frac{1}{|\Sigma|^{\frac{1}{2}}}\exp[-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)] \tag{2.2}$$

其中， $\boldsymbol{x}$ 表示维度为 N 的向量， $\boldsymbol{\mu}$ 表示 $\boldsymbol{x}$ 的均值， Σ 表示向量 $\boldsymbol{x}$ 的协方差矩阵。

当 $N=2$ 时，我们可以得到二元高斯（正态分布）的函数如下：

$$\begin{aligned}&G(x,y;\mu_1,\mu_2,\sigma_1,\sigma_2,\rho)\\&=\frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}}\exp\left\{-\frac{1}{2(1-\rho^2)}\left[\frac{(x-\mu_1)^2}{\sigma_1^2}-\frac{2\rho(x-\mu_1)(y-\mu_2)}{\sigma_1\sigma_2}+\frac{(y-\mu_2)^2}{\sigma_2^2}\right]\right\}\end{aligned} \tag{2.3}$$

其中 ρ 为 x 和 y 的相关系数。

在图像处理中，x、y 相互独立，令式(2.3)中的 $\rho=0$，$\mu_1=\mu_2=0$，$\sigma_1=\sigma_2=\sigma$，这样能获得最终的二元高斯（正态）函数如下：

$$G(x,y;\sigma)=\frac{1}{2\pi\sigma^2}\exp(-\frac{x^2+y^2}{2\sigma^2}) \tag{2.4}$$

为验证其正确性，对其进行二重积分，可得

$$\begin{aligned}\iint G(x,y)dxdy &= \iint \frac{1}{2\pi\sigma^2}\exp(-\frac{x^2+y^2}{2\sigma^2})dxdy \\ &= \int \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{x^2}{2\sigma^2})dx\int \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{y^2}{2\sigma^2})dy = 1\times 1 = 1\end{aligned}$$

因此尺度空间表达式为：

$$L(x,y;\sigma)=G(x,y;\sigma)*I(x,y) \tag{2.5}$$

其中$*$表示卷积运算。尺度空间是自然客观存在的，不是主观创造的。高斯卷积只是尺度空间的一种形式。

二维高斯函数是等高线从中心成正态分布的同心圆。

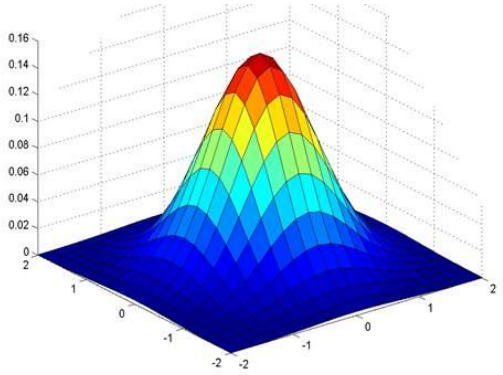


图 2-3 二维高斯函数

分布不为零的点组成卷积阵与原始图像做卷积运算，即每个像素值是周围相邻像素值的高斯平均。一个5×5的高斯模版如下所示：

0.0000 06586	0.0004 24781	0.0133 0373	0.0004 24781	0.0000 06586
0.0004 24781	0.0273 984	0.1098 78	0.0273 984	0.0004 24781
0.0133 0373	0.1098 78	0.4406 55	0.1098 78	0.0133 0373
0.0004 24781	0.0273 984	0.1098 78	0.0273 984	0.0004 24781
0.0000 06586	0.0004 24781	0.0133 0373	0.0004 24781	0.0000 06586

图 2-4 5×5 的高斯模版

高斯模版是圆对称的，且卷积的结果使原始像素值有最大的权重，距离中心越远的相邻像素值权重也越小。在实际应用中，在计算高斯函数的离散近似时，在大概 **3σ** 距离之外的像素都可以看作不起作用，这些像素的计算也就可以忽略。所以，通常程序只计算 $(6\sigma+1)\times(6\sigma+1)$ 就可以保证相关像素影响。

高斯模糊另一个很厉害的性质就是**线性可分**：使用二维矩阵变换的高斯模糊可以通过在水平和竖直方向各进行一维高斯矩阵变换相加得到。

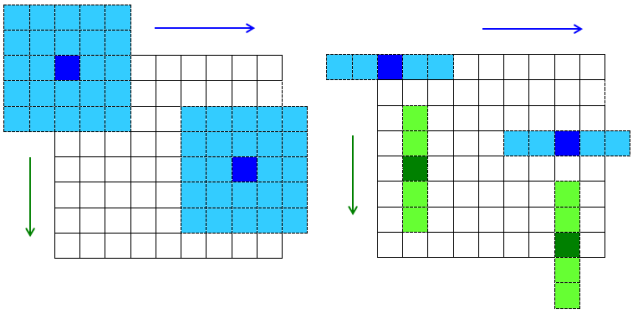


图 2-5 二维高斯卷积使用两个一维高斯卷积替代

$O(N^2\times m\times n)$ 次乘法就能较少成 $O(N\times m\times n)+O(N\times m\times n)$ 次乘法。（N 为高斯核大小，m、n 为二维图像高和宽）

2.5 多尺度和多分辨率

尺度空间表达和金字塔多分辨率表达之间最大的不同是：

- 尺度空间表达是由不同高斯核平滑卷积得到，在所有尺度上有相同的分辨率；
- 而金字塔多分辨率表达每层分辨率减少固定比率。

所以，金字塔多分辨率生成较快，且占用存储空间少；而多尺度表达随着尺度参数的增加冗余信息也变多。

多尺度表达的优点在于图像的局部特征可以用简单的形式在不同尺度上描述；而金字塔表达没有理论基础，难以分析图像局部特征。

第三章 LoG（Laplace of Gaussian）

3.1 高斯拉普拉斯 LoG（Laplace of Gaussian）算子

结合尺度空间表达和金字塔多分辨率表达，就是在使用尺度空间时使用金字塔表示，也就是计算机视觉中最有名的拉普拉斯金子塔（《The Laplacian pyramid as a compact image code》）。

Tony Lindeberg 指出尺度规范化的 LoG(Laplacion of Gaussian)算子具有真正的尺度不变性。高斯拉普拉斯（LoG）算子就是对高斯函数进行拉普拉斯变换形成的卷积算子：

$$\Delta G = \frac{\partial^2 G}{\partial x^2} + \frac{\partial^2 G}{\partial y^2} \tag{3.1}$$

如果我们对式(2.5)形成的尺度空间进行拉普拉斯变换：

$$\Delta L(x, y; \sigma) = \Delta[G(x, y; \sigma) * I(x, y)] \tag{3.2}$$

因为

$$\frac{d}{dt}[h(t) * f(t)] = \frac{d}{dt}\int f(\tau)h(t-\tau)d\tau = \int f(\tau)\frac{d}{dt}h(t-\tau)d\tau = f(t) * \frac{d}{dt}h(t)$$

所以

$$\Delta L(x, y; \sigma) = [\Delta G(x, y; \sigma)] * I(x, y) = \mathrm{LoG} * I(x, y) \tag{3.3}$$

这就形成了最终的 LoG 算子，即使用拉普拉斯算子对高斯函数进行卷积操作。

由于

$$\frac{\partial}{\partial x}G(x, y; \sigma) = \frac{\partial}{\partial x}\left(\frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}}\right) = -\frac{x}{2\pi\sigma^4}e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{3.4}$$

因此

$$\frac{\partial^2}{\partial x^2}G(x, y; \sigma) = \frac{x^2}{2\pi\sigma^6}e^{-\frac{x^2+y^2}{2\sigma^2}} - \frac{1}{2\pi\sigma^4}e^{-\frac{x^2+y^2}{2\sigma^2}} = \frac{x^2-\sigma^2}{2\pi\sigma^6}e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{3.5}$$

对 y 求偏导与对 x 求偏导的形式相同，因此我们可以把 LoG 核函数定义为：

$$\begin{aligned} \mathrm{LoG} = \Delta G(x,y;\sigma) &= \frac{\partial^2}{\partial x^2}G(x,y;\sigma) + \frac{\partial^2}{\partial y^2}G(x,y;\sigma) \\ &= \frac{x^2+y^2-2\sigma^2}{2\pi\sigma^6} e^{-\frac{x^2+y^2}{2\sigma^2}} \end{aligned} \tag{3.6}$$

高斯二阶导如下图的绿色线，蓝色线是高斯一阶导数，红色则是高斯函数。

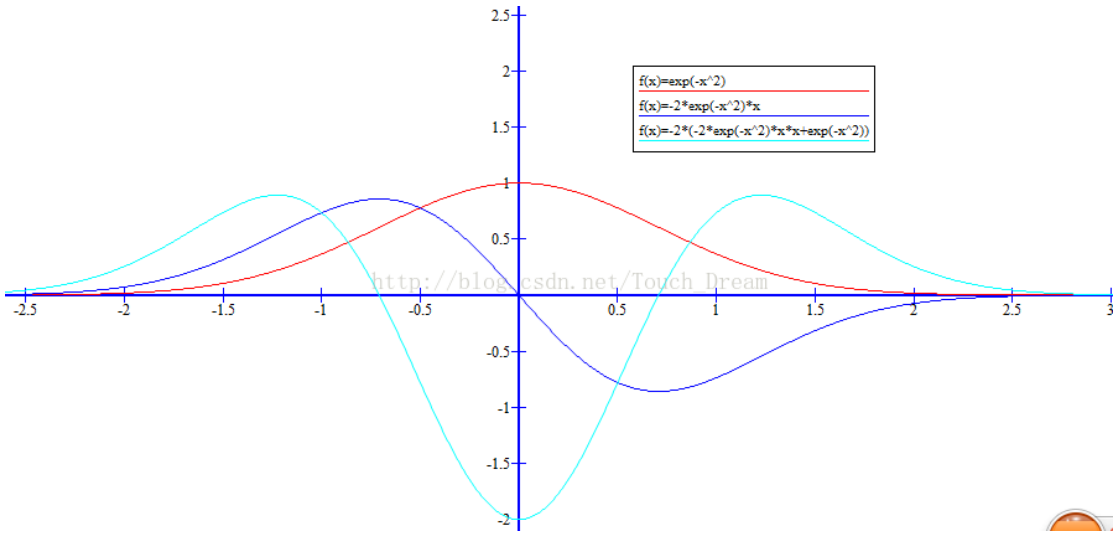


图 3-1 高斯函数及其阶导、二阶导图像

3.2 斑点检测

3.2.1 什么是斑点

斑点通常是指与周围有着颜色和灰度差别的区域。在实际地图中，往往存在着大量这样的斑点，如一颗树是一个斑点，一块草地是一个斑点，一栋房子也可以是一个斑点。由于斑点代表的是一个区域，相比单纯的角点，它的稳定性要好，抗噪声能力要强，所以它在图像配准上扮演了很重要的角色。

同时有时图像中的斑点也是我们关心的区域，比如在医学与生物领域，我们需要从一些 X 光照片或细胞显微照片中提取一些具有特殊意义的斑点的位置或数量。

在视觉领域，斑点检测的主要思路都是检测出图像中比它周围像素灰度值大或比周围灰度值小的区域。一般有两种方法来实现这一目标：

1.基于求导的微分方法，这类方法称为微分检测器

2.基于局部极值的分水岭算法

这里我们重点介绍第一种方法，主要检测 LOG 斑点。利用高斯拉普通拉斯（Laplace of Gaussian，LoG）算子检测图像斑点是一种十分常用的方法。

根据式(3.6)，我们可以获得规范化的高斯拉普拉斯变换为：

$$\Delta_{norm}G = \sigma^2 \Delta G(x, y; \sigma) \tag{3.7}$$

规范化算子在二维图像上显示是一个圆对称函数，如下图所示。我们可以用这个算子来检测图像中的斑点，并且可以通过改变σ的值，可以检测不同尺寸的二维斑点。

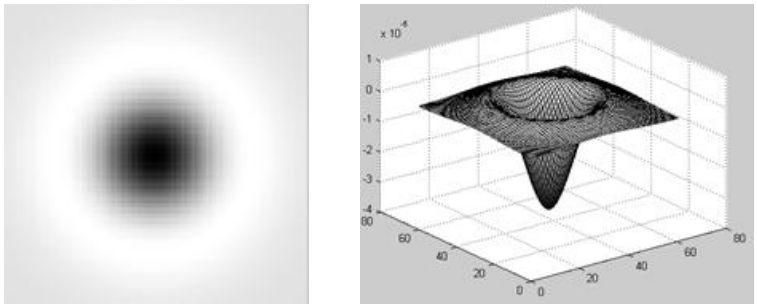


图 3-2 规范化的 LoG 算子在二维图像上的表示

3.2.2 使用 LoG 检测检测斑点原理解释

图像与某一个二维函数进行卷积运算实际就是求取图像与这一函数的相似性。同理，图像与高斯拉普拉斯函数的卷积实际就是求取图像与高斯拉普拉斯函数的相似性。当图像中的斑点尺寸与高斯拉普拉斯函数的形状趋近一致时，图像的拉普拉斯响应达到最大。

事实上我们知道 Laplace 可以用来检测图像中的局部极值点，但是对噪声敏感，所以在我们对图像进行 Laplace 卷积之前，我们用一个高斯低通滤波对图像进行卷积，目标是去除图像中的噪声点。这一过程可以描述为:

先对图像 $f(x, y)$ 用方差为σ的高斯核进行高斯滤波，去除图像中的噪点，与式(2.5)相同，在这里重新写一下公式：

$$L(x, y; \sigma) = G(x, y; \sigma) * I(x, y)$$

然后再根据式(3.2)和式(3.3)对图像使用拉普拉斯算子进行卷积，这里重写下公式：

$$\Delta L(x, y; \sigma) = \Delta[G(x, y; \sigma) * I(x, y)] = [\Delta G(x, y; \sigma)] * I(x, y) = \mathrm{LoG} * I(x, y)$$

这样我们可以获得在**该尺度空间**内的图像的极值点（即斑点）。

我们注意到当σ尺度一定时，只能检测对应半径的斑点，那么检测的是多大半径的斑点呢，我们可以通过对规范化的二维拉普拉斯高斯算子求导获得，由前文可知规范化的高斯拉普拉斯函数为：

$$\Delta_{norm}G=\sigma^2\Delta G(x,y;\sigma)=\frac{x^2+y^2-2\sigma^2}{2\pi\sigma^4}e^{-\frac{x^2+y^2}{2\sigma^2}}$$

将$\Delta_{norm}G$对σ求导，并求其在尺度空间中的极值点：

$$\frac{\partial(\Delta_{norm}G)}{\partial\sigma}=0 \tag{3.8}$$
$$\Rightarrow x^2+y^2-2\sigma^2=0$$

（这个推导好像有问题？）

对于图像中的斑点，在尺度$\sigma=\frac{r}{\sqrt{2}}$时，高斯拉普拉斯响应值达到最大。同理，

如果图像中的圆形斑点黑白反向，那么，它的高斯拉普拉斯响应值在$\sigma=\frac{r}{\sqrt{2}}$时达到最小。将高斯拉普拉斯响应达到峰值时的尺度$\sigma$值，称为特征尺度。如果我们使用不同的$\sigma$值建立 LoG 图，则根据式(3.8)不同的 LoG 图会对不同大小的班点发生相应，如下图所示：

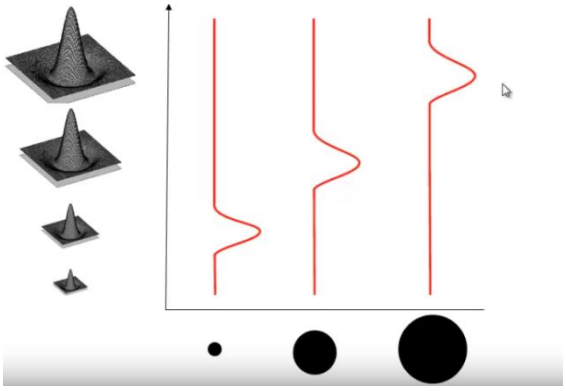


图 3-3 LoG 算子对斑点的相应（K.Grauman，B.Leibe）

图 3-3 左侧是不同尺度参数σ下的 LoG 算子二维图像，其对应着右图的 y 轴。右图 x 轴为不同大小的斑点，纵轴为不同的 LoG 尺度参数，可以看到，针对第一个斑点（即最小的斑点），会在第二个（从下往上数）尺度参数下达到最大响

应，第二个斑点会在第三个尺度参数下达到最大响应，第三个斑点会在第四个尺度下达到最大相应。这是因为随着斑点半径的增大，只有与斑点半径相同的 LoG 算子与斑点进行卷积能够获得最大值，而这个半径唯一对应着一个尺度参数，因此会对该尺度参数在尺度空间内产生极大值，这个图像的结果也与式(3.8)的结果相对应。

在多尺度的情况下，同时在图像空间和尺度空间上达到极值（最大值或最小值）的点就是我们所期望的斑点。这样我们不仅能获得在同一尺度的极值点（斑点），同时还能使获得的斑点成为在连续尺度空间内的极值点。而这些斑点在 SIFT 算法中被定义为图像的关键点。

对于二维图像 $I(x,y)$，计算图像在不同尺度下的离散拉普拉斯响应值，然后检查位置空间中的每个点；如果该点的拉普拉斯响应值都大于或者都小于其他 26 个立方空间领域（9+8+9，图像邻域和尺度邻域）的值，那么该点就是被检测到的图像斑点（即最后的关键点），如图 3-4 所示。

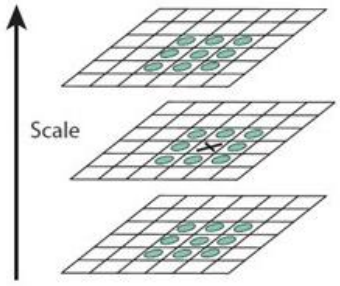


图 3-4 LoG 图在图像空间及尺度空间中寻找极值点

3.3 LoG 斑点检测

下面根据一个图片为例

图 3-5 原图

图 3-5 为所要寻找斑点的原图。下面的图 3-6 为使用不同的尺度空间 σ 参数获得

的 LoG 图像。

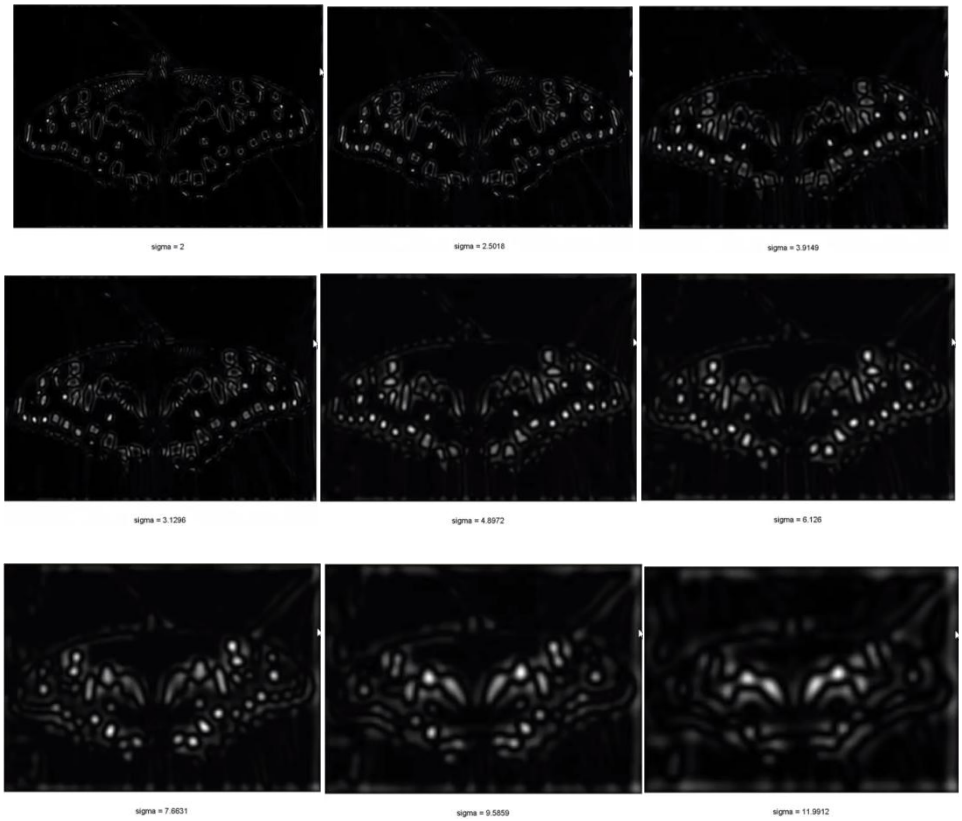


图 3-6 根据不同尺度算子获得的 LoG 图像

根据 3.2.2 节最后介绍的方法，获得 LoG 图像在图像空间以及尺度空间中的极值点作为最终的斑点（关键点）。图 3-7 为最终的结果图。

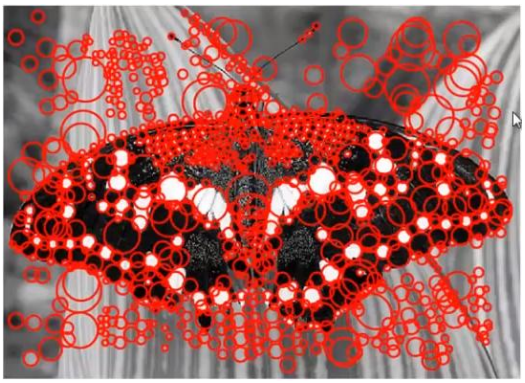


图 3-7 LoG 斑点提取结果图

图中斑点的圆心为所获得的斑点（关键点），根据圆的半径以及式(3.8)可获得其对应尺度参数，即该斑点是在通过半径计算得到的尺度参数下的尺度空间内获得的极值点，既为图像空间极值点也是尺度空间极值点。

第四章 DoG（Difference of Gaussian）

4.1 高斯差分（DoG）算子

如果我们将二维高斯函数对尺度参数σ求导，则可以得到：

$$\begin{aligned}\frac{\partial}{\partial\sigma}G(x,y;\sigma)&=\frac{\partial}{\partial\sigma}[\frac{1}{2\pi\sigma^2}\exp(-\frac{x^2+y^2}{2\sigma^2})]\\&=-\frac{1}{\pi\sigma^3}\exp(-\frac{x^2+y^2}{2\sigma^2})+\frac{1}{2\pi\sigma^2}\frac{x^2+y^2}{\sigma^3}\exp(-\frac{x^2+y^2}{2\sigma^2})\\&=\frac{x^2+y^2}{2\pi\sigma^5}\exp(-\frac{x^2+y^2}{2\sigma^2})-\frac{1}{\pi\sigma^3}\exp(-\frac{x^2+y^2}{2\sigma^2})\\&=\frac{x^2+y^2-2\sigma^2}{2\pi\sigma^5}\exp(-\frac{x^2+y^2}{2\sigma^2})\end{aligned}\tag{4.1}$$

由上文的式(3.6)可知

$$\mathrm{LoG}=\Delta G(x,y;\sigma)=\frac{x^2+y^2-2\sigma^2}{2\pi\sigma^6}e^{-\frac{x^2+y^2}{2\sigma^2}}$$

因此

$$\frac{\partial}{\partial\sigma}G(x,y;\sigma)=\sigma\Delta G(x,y;\sigma)\tag{4.2}$$

根据导数的定义可得

$$\begin{aligned}\frac{\partial}{\partial\sigma}G(x,y;\sigma)&=\lim_{\Delta\sigma\to0}\frac{G(x,y;\sigma+\Delta\sigma)-G(x,y;\sigma)}{(\sigma+\Delta\sigma)-\sigma}\\&\approx\frac{G(x,y;k\sigma)-G(x,y;\sigma)}{k\sigma-\sigma}\end{aligned}\tag{4.3}$$

所以

$$\sigma\Delta G(x,y;\sigma)=\frac{\partial}{\partial\sigma}G(x,y;\sigma)\approx\frac{G(x,y;k\sigma)-G(x,y;\sigma)}{k\sigma-\sigma}\tag{4.4}$$

变形一下得到：

$$G(x,y;k\sigma)-G(x,y;\sigma)=(\mathrm{k}-1)\sigma^2\Delta G(x,y;\sigma)\tag{4.5}$$

等式右边比 LoG 算子只是多一个系数，在实际应用中不受影响。

因此我们定义：

$$DoG=G(x,y;k\sigma)-G(x,y;\sigma)\tag{4.6}$$

用以代替 LOG 算子与图像卷积。好处是可以提高算法的效率，减少计算量。

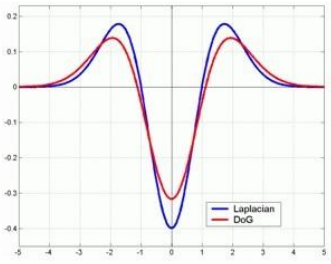


图 4-1 高斯拉普拉斯算子和高斯差分的比较

从两个平平滑算子的差分得出的是二阶边缘检测，反直观。近似计算可能如下图所示。图中一维空间，不同变量的两个高斯分布相减形成一个一维算子

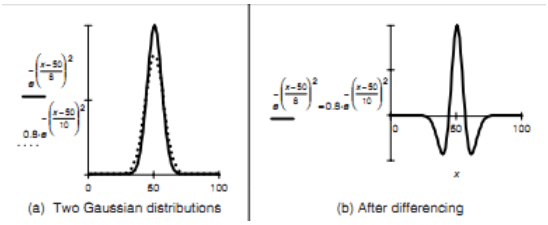


图 4-2 一维高斯分布差分

注意最后计算的模板算子的权重和应该保证为 1，不是 1 的可以进行归一化！

4.2 高斯差分 DoG 金字塔构建

由第三章，我们知道使用 LoG 算子对图像进行卷积获得的是图像中的斑点，斑点的半径对应不同的尺度参数 σ。SIFT 算法建议，在某一尺度上的特征检测可以通过对两个相邻高斯尺度空间的图像相减，得到 DoG 的响应值图像 $D(x,y;\sigma)$。因此可以仿照 LoG 方法，通过对响应值图像 $D(x,y;\sigma)$ 进行局部最大值搜索，在空间位置和尺度空间定位局部关键点。其中：

$$\begin{aligned} D(x,y;\sigma) &= (G(x,y;k\sigma) - G(x,y;\sigma)) * I(x,y) \\ &= L(x,y;k\sigma) - L(x,y;\sigma) \end{aligned} \tag{4.7}$$

k 为相邻两个尺度空间倍数的常数。

通过 4.1 节的推导可知，DoG（Difference of Gaussian）其实是对高斯拉普拉斯 LoG 的近似，也就是对 $\sigma^2 \Delta G(x,y;\sigma)$ 的近似。因此我们在这一节获得的 DoG 图像实际上也是斑点的图像，所得的斑点的半径对应通过 DoG 图近似获得的响应 LoG 图的尺度参数 σ，近似由式(4.5)所得。

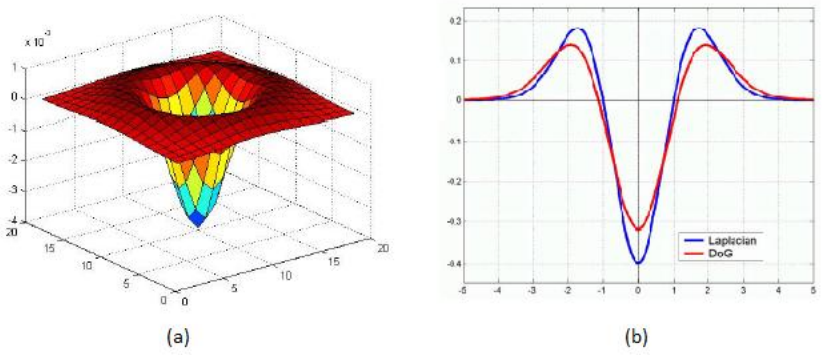


图 4-3 （a）是 DoG 的三维图，（b）是 DoG 与 LoG 的对比

4.2.1 构建高斯金字塔

为了得到 DoG 图像，先要构造高斯金字塔。我们回过头来继续说高斯金字塔。

高斯金字塔在多分辨率金字塔简单**降采样基础上加了高斯滤波**，也就是对金字塔每层图像用不同参数的σ做高斯模糊，使得每层金字塔有多张高斯模糊图像。金字塔每层多张图像合称为一组（Octave），每组有多张（也叫层 Interval）图像。

金字塔的组数根据图像的原始大小和塔顶图像的大小共同决定，其计算公式如下：

$$n = \log_2[\min(M, N)] - t \tag{4.8}$$

其中 M、N 为原图像的长和宽，t 为塔顶图像的最小维数的对数值，t 一般取 3~5。

采样与尺度参数之间的关系

我们现在知道了在高斯金字塔中，同一组的图像是根据不同的尺度参数由高斯函数进行卷积获得的，同一组图像在尺度空间中可用高斯函数的尺度参数来表示；不同组的图像之间是通过降采样获得的，那么降采样与尺度空间的关系是什么呢？

一下分析来源于我自己的猜测，具体关系还需要查看尺度空间的相关文献。首先对图像进行降采样（缩小）或放大对图像的尺度没有影响，以降采样为例，在直觉上，我们可能认为降采样会使图像尺度变小，实际上，降采样会导致我们丢失图像中的部分信息丢失，这会给我们一种尺度变小的感觉，但实际上这种信息的丢失与尺度变换中的模糊是不同的。在某一个参考资料中是这么解释这个问

题的，我觉得很有道理：尺度相对于尺寸（图片大小，像素点的数量）才有意义。比如，一张图片扩大两倍以后用 3.2 的尺度参数进行高斯模糊，和对原图片用 1.6 的尺度参数进行模糊，是相同的效果。这里对这个问题进行简单的证明。首先考虑初始的图像 $I_1(x,y)$，我们使用尺度参数为 σ_1 的高斯函数对它进行卷积，随后考虑图像 $I_2(x,y)$，它是 $I_1(x,y)$ 缩小一半后获得的图像。显然当图片某点距离原点距离 n，放大 k 倍以后的图片距离为 k*n。正态分布公式中，距离的平方除以方差（这里指的是尺度）的平方，因此放大以后图片为了相似的权重效果也需要放大尺度为 k 倍。

对于 SIFT 算法来说，尺度空间的尺度参数是对于原图像 $I_s(x,y)$ 的图片尺寸来说的。

为了与 OpenCV 源码相对应，因此设定金字塔最底下的组别号为 0，每组中的首层为第 0 层。

高斯金字塔第 0 组第 0 层图像的生成

很多初涉 SIFT 的都会被这个问题所困惑，这里要分两种情况：其一是把第 0 组的索引定为 0；其二是把第 0 组的索引定为-1。

我们先考虑第 0 组索引为 0 的情况，（在 Low 的论文中）我们知道第 0 组第 0 层的图像是由原始图像与 σ_0（一般设置为 1.6）的高斯滤波器卷积生成，那么原始图像是谁呢？是 $I(x,y)$ 吗？不是！为了图像反走样的需要，通常假设输入图像是经过高斯平滑处理的，其值为 $\sigma_n=0.5$，即半个像元。意思就是说我们采集到的输入图像 $I(x,y)$，并不是原始图像，它已经被 $\sigma=\sigma_n=0.5$ 的高斯滤波器平滑过了（为了防止混淆效应），在这里我们定义这个未被 $\sigma=\sigma_n=0.5$ 处理过的原始图像为 $I_s(x,y)$，它其实是一个具有无限分辨率的图像，也就是说它的尺度 $\sigma=0$，但是这样的图像实际无法获得。另外再考虑一个问题，这个原始图像 $I_s(x,y)$ 的尺寸是多大？其实可以是任意大小，首先因为我们假定它的分辨率是无限，其次对图像进行放大缩小不会对图像的尺度产生影响。综上，获取第 0 组第 0 层图像的时候，我们不能直接对输入图像 $I(x,y)$ 用 σ_0 的高斯滤波器平滑，

而应该用 $\sigma=\sqrt{\sigma_0^2-\sigma_n^2}$（高斯函数的可加性，后文有证明）的高斯滤波器去平滑输入图像 $I(x,y)$，即

$$FirstLayer(x,y)=I(x,y)*G(x,y;\sqrt{\sigma_0^2-\sigma_n^2}) \tag{4.9}$$

其中 $FirstLayer(x,y)$ 表示整个尺度空间第 0 组第 0 层的图像，σ_0 一般取 1.6，$\sigma_n=0.5$。

这里式(4.9)的推导如下：

对于输入图像 $I(x,y)$，它与原始图像 $I_s(x,y)$ 的关系如下：

$$I(x,y)=I_s(x,y)*G(\mathrm{x},\mathrm{y};\sigma_n)$$

而由前文的叙述可知，高斯金字塔（尺度空间）的第 0 组第 0 层图像应该是原始图像用 σ_0 的高斯滤波器平滑后得到的图像，因此第 0 组第 0 层图像与原始图像 $I_s(x,y)$ 间的关系为

$$FirstLayer(x,y)=I_s(x,y)*G(\mathrm{x},\mathrm{y};\sigma_0)$$

因此我们可以认为第 0 组第 0 层的图像在尺度空间的位置为 σ_0（相对于原始图像 $I_s(x,y)$ 来说）。我们现在需要获得第 0 组第 0 层图像 $FirstLayer(x,y)$ 与输入图像 $I(x,y)$ 的关系，因为实际上并没有原始图像 $I_s(x,y)$，我们只能对输入图像进行处理。我们希望他们之间的关系也可以一个用高斯函数进行卷积的关系来表示，因此定义它们间的关系为

$$FirstLayer(x,y)=I(x,y)*G(\mathrm{x},\mathrm{y};\sigma)$$

将输入图像 $I(x,y)$ 与原始图像 $I_s(x,y)$ 的关系带入，可以得到如下关系

$$FirstLayer(x,y)=I(x,y)*G(\mathrm{x},\mathrm{y};\sigma)=I_s(x,y)*G(\mathrm{x},\mathrm{y};\sigma_n)*G(\mathrm{x},\mathrm{y};\sigma)$$

（查阅相关资料，卷积具有如下代数性质：

1.交换律 $x(t)*h(t)=h(t)*x(t)$

2.分配律 $x(t)*[h_1(t)+h_2(t)]=x(t)*h_1(t)+x(t)*h_2(t)$

3.结合律$[x(t)*h_1(t)]*h_2(t)=x(t)*[h_1(t)*h_2(t)]$

以上性质不在这里进行证明，证明过程可查阅相关资料。）

根据卷积的结合律，可得到以下关系

$$FirstLayer(x,y)=I_s(x,y)*G(\mathrm{x},\mathrm{y};\sigma_n)*G(\mathrm{x},\mathrm{y};\sigma)=I_s(x,y)*[G(\mathrm{x},\mathrm{y};\sigma_n)*G(\mathrm{x},\mathrm{y};\sigma)]$$

（推导：两个高斯函数卷积后仍是高斯函数

首先定义两个高斯函数

$$g_1(x)=\frac{1}{\sqrt{2\pi}\sigma_1}e^{-\frac{x^2}{2\sigma_1^2}},g_2(x)=\frac{1}{\sqrt{2\pi}\sigma_2}e^{-\frac{x^2}{2\sigma_2^2}}$$

$f(t)$为这两个高斯函数的卷积，则 $f(t)$ 表示如下

$$f(t)=g_1(x)*g_2(x)=\int g_1(x)\times g_2(t-x)dx$$

$f(t)$的具体推导过程如下：

$$f(t)=\int\frac{1}{\sqrt{2\pi}\sigma_1}e^{-\frac{x^2}{2\sigma_1^2}}\times\frac{1}{\sqrt{2\pi}\sigma_2}e^{-\frac{(t-x)^2}{2\sigma_2^2}}dx=\frac{1}{2\pi\sigma_1\sigma_2}\int e^{-\frac{x^2}{2\sigma_1^2}-\frac{(t-x)^2}{2\sigma_2^2}}dx$$

$$=\frac{1}{2\pi\sigma_1\sigma_2}\int e^{-\frac{2\sigma_2^2x^2+2\sigma_1^2(t-x)^2}{4\sigma_1^2\sigma_2^2}}dx$$

$$=\frac{1}{2\pi\sigma_1\sigma_2}\int e^{-\frac{(\sigma_1^2+\sigma_2^2)\left(x^2-\frac{2\sigma_1^2}{\sigma_1^2+\sigma_2^2}\right)+\sigma_1^2t^2}{2\sigma_1^2\sigma_2^2}}dx$$

$$=\frac{1}{2\pi\sigma_1\sigma_2}\int e^{-\frac{(\sigma_1^2+\sigma_2^2)\left(x-\frac{\sigma_1^2}{\sigma_1^2+\sigma_2^2}t\right)^2-\frac{\sigma_1^4}{\sigma_1^2+\sigma_2^2}t^2+\sigma_1^2t^2}{2\sigma_1^2\sigma_2^2}}dx$$

$$=\frac{1}{2\pi\sigma_1\sigma_2}\int e^{-\frac{(\sigma_1^2+\sigma_2^2)\left(x-\frac{\sigma_1^2}{\sigma_1^2+\sigma_2^2}t\right)^2-\frac{\sigma_1^2\sigma_2^2}{\sigma_1^2+\sigma_2^2}t^2}{2\sigma_1^2\sigma_2^2}}dx$$

$$=\frac{1}{2\pi\sigma_1\sigma_2}\int e^{-\frac{(\sigma_1^2+\sigma_2^2)\left(x-\frac{\sigma_1^2}{\sigma_1^2+\sigma_2^2}t\right)^2}{2\sigma_1^2\sigma_2^2}}\times e^{-\frac{1}{2(\sigma_1^2+\sigma_2^2)}t^2}dx$$

$$=\frac{1}{2\pi\sigma_1\sigma_2}e^{-\frac{1}{2(\sigma_1^2+\sigma_2^2)}t^2}\int e^{-\frac{(\sigma_1^2+\sigma_2^2)\left(x-\frac{\sigma_1^2}{\sigma_1^2+\sigma_2^2}t\right)^2}{2\sigma_1^2\sigma_2^2}}dx$$

$$=\frac{1}{2\pi\sigma_1\sigma_2}e^{-\frac{1}{2(\sigma_1^2+\sigma_2^2)}t^2}\times A$$

其中 $A=\int e^{-\frac{(\sigma_1^2+\sigma_2^2)\left(x-\frac{\sigma_1^2}{\sigma_1^2+\sigma_2^2}t\right)^2}{2\sigma_1^2\sigma_2^2}}dx$，它的结果为一个常数，下面给出计算过程：

$$A=\int e^{-\frac{(\sigma_1^2+\sigma_2^2)\left(x-\frac{\sigma_1^2}{\sigma_1^2+\sigma_2^2}t\right)^2}{2\sigma_1^2\sigma_2^2}}dx$$

运用因式替换计算，首先令 $y=x-\frac{\sigma_2^2}{\sigma_1^2+\sigma_2^2}t$，则可知 $dy=dx$，代入上式中可得：

$$A=\int e^{-\frac{(\sigma_1^2+\sigma_2^2)y^2}{2\sigma_1^2\sigma_2^2}}dy$$

为了计算 A，将 A 用另外一种形式表示：

$$A=\int e^{-\frac{(\sigma_1^2+\sigma_2^2)x^2}{2\sigma_1^2\sigma_2^2}}dx$$

这样

$$A\times A=\int e^{-\frac{(\sigma_1^2+\sigma_2^2)x^2}{2\sigma_1^2\sigma_2^2}}dx\times\int e^{-\frac{(\sigma_1^2+\sigma_2^2)y^2}{2\sigma_1^2\sigma_2^2}}dy=\iint e^{-\frac{(\sigma_1^2+\sigma_2^2)(x^2+y^2)}{2\sigma_1^2\sigma_2^2}}dxdy$$

可用下式将上面的积分转换为极坐标形式：

$$\begin{cases}x=r\times\cos\theta\\ y=r\times\sin\theta\\ dxdy=rdrd\theta\end{cases}$$

这样就可以得到

$$A\times A=A^2=\int_0^{2\pi}\int_0^{+\infty}\int e^{-\frac{(\sigma_1^2+\sigma_2^2)r^2}{2\sigma_1^2\sigma_2^2}}rdrd\theta=\int_0^{2\pi}1d\theta\int_0^{+\infty}e^{-\frac{(\sigma_1^2+\sigma_2^2)r^2}{2\sigma_1^2\sigma_2^2}}rdr$$

$$=2\pi\left[-\frac{\sigma_1^2\sigma_2^2}{(\sigma_1^2+\sigma_2^2)}e^{\frac{(\sigma_1^2+\sigma_2^2)r^2}{2\sigma_1^2\sigma_2^2}}\right]_0^{-\infty}=2\pi\frac{\sigma_1^2\sigma_2^2}{(\sigma_1^2+\sigma_2^2)}$$

由此可知，A 的值为

$$A=\sqrt{2\pi}\frac{\sigma_1\sigma_2}{\sqrt{(\sigma_1^2+\sigma_2^2)}}$$

由此可得

$$f(t)=\frac{1}{2\pi\sigma_1\sigma_2}Ae^{-\frac{1}{2(\sigma_1^2+\sigma_2^2)}t^2}=\frac{1}{\sqrt{2\pi}\sqrt{(\sigma_1^2+\sigma_2^2)}}e^{-\frac{1}{2(\sigma_1^2+\sigma_2^2)}t^2}$$

$$=\frac{1}{\sqrt{2\pi}\sigma_3}e^{-\frac{t^2}{2\sigma_3^2}}$$

其中 $\sigma_3^2=\sigma_1^2+\sigma_2^2$，由此可知两个高斯函数卷积仍为高斯函数，新高斯函数的方差为原来两个高斯函数方差的和。

对相互独立的两个二元高斯函数，推导过程类似，可以将二元高斯函数分离成两个一元高斯函数，进而进行推导，因此结论相同。）

因此可以如下关系

$$G(\mathrm{x},\mathrm{y};\sigma_0)=G(\mathrm{x},\mathrm{y};\sigma_n)*G(\mathrm{x},\mathrm{y};\sigma)$$

$$\sigma_0^2=\sigma_n^2+\sigma^2$$

因此可以得到第 0 组第 0 层图像 $FirstLayer(x,y)$ 与输入图像 $I(x,y)$ 的关系为

$$FirstLayer(x,y)=I(x,y)*G(\mathrm{x},\mathrm{y};\sigma)$$

其中 $\sigma=\sqrt{\sigma_0^2-\sigma_n^2}$ 。

现在我们考虑第 0 组的索引为-1 的情况。那么首先第一个问题便是为什么要把索引定为-1；如果索引为 0，那么整个尺度空间的第 0 组的第 0 层图像已经是由输入图像模糊生成的了，也就是说已经丢失了细节部分信息，输入图像的信息我们没有完全利用上。基于这种考虑，我们先将图像放大 2 倍，这样原图像的细节就隐藏在了放大后的图像中。

由上文对第一种情况的分析可知，我们已经把输入图像 $I(x,y)$ 看成是已经被 $\sigma_n=0.5$ 模糊过的图像，那么将 $I(x,y)$ 放大 2 倍后得到 $I_n(x,y)$，则可以看为是原始图像 $I_s(x,y)$ 被 $\sigma=2\sigma_n=1$ 的高斯核模糊过的图像（根据前文采样与尺度参数之间的关系）。那么由 $I_n(x,y)$ 生成的第 0 组第 0 层图像应该使用 $\sigma=\sqrt{\sigma_0^2-(2\sigma_n)^2}$ 的高斯滤波器产生，即

$$FirstLayer(x,y)=I_n(x,y)*G(\mathrm{x},\mathrm{y};\sqrt{\sigma_0^2-(2\sigma_n)^2})$$

其中 $I_n(x,y)$ 为输入图像 $I(x,y)$ 放大 2 倍后得到图像，第 0 组第 0 层图像的尺度空间表示为 $L(x,y;\sigma_0)$（相对于原始图片 $I_s(x,y)$ 来说）。

高斯金字塔第 0 组剩余图像的生成

由上文的讨论可以知道，我们认为第 0 组第 0 层图像相对于原始图像的尺度为 $\sigma_{0,0}=\sigma_0=1.6$，即它是对原始图像使用 $\sigma=\sigma_{0,0}=\sigma_0=1.6$ 的高斯函数进行卷积获得的（$\sigma_{o,s}$ 表示第 o 组第 s 层图像相对于原始图像 $I_s(x,y)$ 的尺度参数，$o=0,1,2,\cdots$，$s=0,1,2,\cdots$）。根据式(4.4)、式(4.5)和式(4.7)，我们可以知道对图像使用 $G(x,y;\sigma)$ 的高斯函数进行卷积可以获得尺度图像 $L(x,y;\sigma)$，使用相邻两个尺度图像 $L(x,y;k\sigma)$ 和 $L(x,y;\sigma)$ 进行查分可以获得 LoG 图像的近似图像 DoG 图像，我们最终希望获得的是 DoG 图像的金字塔，而高斯金字塔中的图像都可以认为是对原始图像使用了高斯函数进行卷积后的图像，因此尺度图像 $L(x,y;\sigma)$ 其实就是高斯金字塔中的图像。因此对某一层的高斯金字塔来说，若第 0 层图像为尺度图像 $L(x,y;\sigma)$，则第 1 层图像为尺度图像 $L(x,y;k\sigma)$，第 2 层图像为尺度图像 $L(x,y;k^2\sigma)$，第 3 层图像为尺度图像 $L(x,y;k^3\sigma)$，该组的其他图像以此类推，该组第 s 层（$s=0,1,2\cdots$）图像即为尺度图像 $L(x,y;k^s\sigma)$。

尺度空间里的每一层的图像（除了第 0 层）都可以由其前面一层的图像和一个相对尺度参数为 σ 的高斯滤波器卷积生成，而不是由原图和对应尺度的高斯滤波器生成的，这一方面是因为我前面提到的不存在所谓意思上的“原图”，我们的输入图像 $I(x,y)$ 已经是尺度为 $\sigma=\sigma_n=0.5$ 的图像了。另一方面是由于如果用原图计算，那么相邻两层之间相差的尺度实际上非常小，这样会造成在做高斯差分图像的时候，大部分值都趋近于 0，以致于后面我们很难检测到关键点。

基于上面两点原因，所以**对于每一组的第 i+1 层的图像，都是由第 i 层的图像和一个相对尺度的高斯滤波器卷积生成**。

针对第 0 组的剩余图像，根据上文的叙述可以这样获得。首先已经确定的是

第 0 组第 0 层图像是对输入图像 $I(x,y)$ 放大两倍得到 $I_n(x,y)$ 后，使用 $\sigma=\sqrt{\sigma_0^2-(2\sigma_n)^2}$ 的高斯函数对 $I_n(x,y)$ 进行卷积产生的。相对于原始图像，它是尺度图像 $L(x,y;\sigma_{0,0}=\sigma_0=1.6)$。对于第 0 组的第 1 层图像，由前文叙述我们希望它是尺度图像 $L(x,y;\sigma_{0,1}=k\sigma_{0,0})$，由于高斯函数卷积的可加性（证明在前文），可以得到如下推导：

$$\begin{aligned}L(x,y;\sigma_{0,1}=k\sigma_{0,0}=k\sigma_0)&=I_s(x,y)*G(x,y;\sigma_{0,1})\\&=I_s(x,y)*G(x,y;k\sigma_{0,0})\\&=I_s(x,y)*G(x,y;\sigma_{0,0})*G(x,y;\sigma_{0,diff\,0})\\&=L(x,y;\sigma_{0,0}=\sigma_0=1.6)*G(x,y;\sigma_{0,diff\,0})\end{aligned}\tag{4.10}$$

其中 $\sigma_{0,diff\,0}$ 可通过如下推导获得

$$\begin{aligned}&\sigma_{0,1}^2=\sigma_{0,0}^2+\sigma_{0,diff\,0}^2\\\Rightarrow\ &\sigma_{0,diff\,0}=\sqrt{\sigma_{0,1}^2-\sigma_{0,0}^2}\\\Rightarrow\ &\sigma_{0,diff\,0}=\sqrt{(k\sigma_{0,0})^2-\sigma_{0,0}^2}\end{aligned}\tag{4.11}$$

因此第一组第二层图像可以通过对第一组第一层图像使用高斯函数 $G(x,y,\sigma_{0,diff\,0})$ 获得，其中 $\sigma_{0,diff\,0}=\sqrt{(k\sigma_{0,0})^2-\sigma_{0,0}^2}$，$k$ 的值暂时未知，在后文中进行推导，它的尺度空间表示为 $L(x,y;k\sigma_0)$（相对于原始图像 $I_s(x,y)$ 来说）。

对于高斯金字塔第 0 组的第 2 层图像可以通过类似式(4.10)的推导，获得如下关系

$$\begin{aligned}L(x,y;\sigma_{0,2}=k^2\sigma_{0,0}=k^2\sigma_0)&=I_s(x,y)*G(x,y;\sigma_{0,2})\\&=I_s(x,y)*G(x,y;k^2\sigma_{0,0})\\&=I_s(x,y)*G(x,y;k\sigma_{0,0})*G(x,y;\sigma_{0,diff\,0})\\&=L(x,y;\sigma_{1,0}=k\sigma_0)*G(x,y;\sigma_{0,diff\,1})\end{aligned}\tag{4.12}$$

其中 $\sigma_{0,diff\,1}$ 可通过如下推导获得

$$\begin{aligned}&\sigma_{0,2}^2=\sigma_{0,1}^2+\sigma_{0,diff\,1}^2\\\Rightarrow\ &\sigma_{0,diff\,1}=\sqrt{\sigma_{0,2}^2-\sigma_{0,1}^2}\\\Rightarrow\ &\sigma_{0,diff\,1}=\sqrt{(k^2\sigma_{0,0})^2-(k\sigma_{0,0})^2}\end{aligned}\tag{4.13}$$

因此第 0 组第 2 层图像可以通过对第 0 组第 1 层图像使用高斯函数 $G(\mathrm{x},\mathrm{y},\sigma_{0,diff1})$ 获得，其中 $\sigma_{0,diff1}=\sqrt{(k^2\sigma_{0,0})^2-(k\sigma_{0,0})^2}$，它的尺度空间表示为 $L(x,y;k^2\sigma_0)$（相对于原始图像 $I_s(x,y)$ 来说）。

依次类推，第 0 组第 3 层，可以通过对第 0 组第 2 张层图像使用高斯函数 $G(\mathrm{x},\mathrm{y},\sigma_{0,diff2})$ 获得，其中 $\sigma_{0,diff2}=\sqrt{(k^3\sigma_{1,1})^2-(k^2\sigma_{1,1})^2}$。它的尺度空间表示为 $L(x,y;k^3\sigma_0)$（相对于原始图像 $I_s(x,y)$ 来说）。

综上，第 0 组第 s 层，可以通过对第 0 组第 s-1 层图像使用高斯函数 $G(\mathrm{x},\mathrm{y},\sigma_{0,diff(s-1)})$ 获得，其中 $\sigma_{0,diff(s-1)}=\sqrt{(k^s\sigma_{1,1})^2-(k^{s-1}\sigma_{1,1})^2}$

高斯金字塔第 1 组图像的生成

由前文已知，高斯金字塔不同组的第 0 层图像是通过对前一组的尺度空间（高斯）图像降采样获得的，那具体是对前一组的那一层图片进行降采样获得的呢？是对前一组的第一层图像吗？还是对前一组的其他层图像进行降采样获得的？

首先我们来看看第 0 组图像的尺度（都是相对于原始图像 $I_s(x,y)$ 的尺度，而不是相对于输入图像 $I(x,y)$ 的尺度）：

组数（Octave）	层数（Interval）	尺度空间表示（相对于 $I_s(x,y)$ ）	尺度参数	相对于前一层的相对尺度参数 $\sigma_{0,diff(n-1)}$
0	0	$L(x,y;\sigma_0)$	σ_0	σ_0（相对于原始图像的尺度）
	1	$L(x,y;k\sigma_0)$	$k\sigma_0$	$\sqrt{(k\sigma_{0,0})^2-\sigma_{0,0}^2}$
	2	$L(x,y;k^2\sigma_0)$	$k^2\sigma_0$	$\sqrt{(k^2\sigma_{0,0})^2-(k\sigma_{0,0})^2}$
	3	$L(x,y;k^3\sigma_0)$	$k^3\sigma_0$	$\sqrt{(k^3\sigma_{0,0})^2-(k^2\sigma_{0,0})^2}$
	4	$L(x,y;k^4\sigma_0)$	$k^4\sigma_0$	$\sqrt{(k^4\sigma_{0,0})^2-(k^3\sigma_{0,0})^2}$

	⋮	⋮	⋮	⋮
	s	$L(x,y;k^{n-1}\sigma_0)$	$k^{s-1}\sigma_0$	$\sqrt{(k^{s-1}\sigma_{0,0})^2-(k^{s-2}\sigma_{0,0})^2}$

我们希望**相邻两组的同一层尺度为 2 倍的关系**。因此对于第二组的高斯金字塔来说，第 1 组第 0 层图像的尺度空间表示应该是 $L(x,y;2\sigma_0)$（相对于原始图像 $I_s(x,y)$ 来说），尺度参数为 $\sigma_{1,0}=2\sigma_0$ ，对第二组仍使用相同的 k，则第 1 组第 1 层图像的尺度空间表示应该是 $L(x,y;k\times 2\sigma_0)$，其相对第 1 组第 0 层的尺度为 $\sqrt{(k\times 2\sigma_{1,0})^2-(2\sigma_{1,0})}=2\sqrt{(k\sigma_0)^2-(\sigma_0)}=2\sigma_{0,diff\,0}$；第 1 组第 2 层图像的尺度空间表示应该是 $L(x,y;k^2\times 2\sigma_0)$，其相对第 1 组第 1 层的尺度为 $\sqrt{(k^2\times 2\sigma_{1,0})^2-(k\times 2\sigma_{1,0})}=2\sqrt{(k^2\sigma_0)^2-(k\sigma_0)}=2\sigma_{0,diff\,1}$，第 1 组之其他层的图像依此类推。这样第 1 组每层的相对尺度应该是第 0 组的 2 倍，后面其他组的相对尺度也应该与第一组不同。首先，如果第 1 组图像的大小与第 0 组图像的大小相同的话，这个推导是没有问题的，但是第 1 组的图像是在第 0 组某层的基础上降采样获得的，即第 1 组的图像的 height 和 width 是第 0 组的一半。我们从前文已经知道，降采样对图像的尺度是没有影响的，因此在获取第 1 组第 0 层图像时，应该对第一组中尺度参数为 $\sigma_{0,i}=k^i\sigma_0=2\sigma_0$ 的第 i 层图像进行降采样获得，这样能保证第 1 组第 0 层图像的尺度参数是第 0 组第 0 层图像的尺度参数的 2 倍。但是这里有一个问题，前文我们提到过，单独说尺度是没有意义的，尺度相对于尺寸（图片大小，像素点的数量）才有意义，那这里的尺度是相对于多大尺寸的图像呢？是相对于第 0 组图像尺寸大小的图像吗？我们回到第 0 组第 0 层图像，它的获取方法如下：首先将输入图像 $I(x,y)$ 放大两倍后得到 $I_n(x,y)$，由于输入图像 $I(x,y)$ 可以看成是原始图像 $I_s(x,y)$ 被 $\sigma=\sigma_n=0.5$ 的高斯核模糊过的图像，因此经过放大后的 $I_n(x,y)$，可以看作是原始图像 $I_s(x,y)$ 被 $\sigma=2\sigma_n=1$ 的高斯核模糊过的图像（根据前文采样与尺度参数之间的关系）。那么由 $I_n(x,y)$ 生成的第 0 组第 0 层图像应该使用 $\sigma=\sqrt{\sigma_0^2-(2\sigma_n)^2}$ 的高斯滤波器产生，即

$$FirstLayer(x,y)=I_n(x,y)*G(\mathrm{x},\mathrm{y};\sqrt{\sigma_0^2-(2\sigma_n)^2})$$

其中 $I_n(x,y)$ 为输入图像 $I(x,y)$ 放大 2 倍后得到图像，因此第 0 组第 0 层图像的尺度空间表示为 $L(x,y;\sigma_0)$，这个尺度空间表示是相对于原始图片 $I_s(x,y)$ 来说的，但是同样的，是它相对于对大尺寸的 $I_s(x,y)$ 来说的呢？这里我理解是相对与第 0 组图像尺寸大小相同的 $I_s(x,y)$ 来说的，首先 $I_n(x,y)$ 是通过输入图像 $I(x,y)$ 放大获得的，因此 $I_n(x,y)$ 和 $I(x,y)$ 的尺度应该是相同的，但是在推导过程中我们认为 $I_n(x,y)$ 是相对于 $I_s(x,y)$ 的尺度为 $2\sigma_n$，而 $I(x,y)$ 相对于 $I_s(x,y)$ 的尺度为 σ_n，这说明，描述 $I_n(x,y)$ 的尺度时，是基于与它尺寸大小相同的 $I_s(x,y)$ 描述的。因此在叙述第 0 组的其他层时，也是基于与 $I_n(x,y)$ 尺寸大小相同的 $I_s(x,y)$ 描述的。因此在描述第 1 组图像的尺度时，我们也是基于同样的 $I_s(x,y)$ 描述的。但是第 1 组的图像大小变为了第 0 组的一半，在获取第 1 组第 0 层图像时，我们仅仅是通过对第 0 组尺度为 $2\sigma_0$ 的图像降采样获得，因此第 1 组第 0 层的尺度没有疑问，与降采样前的图像相同。但是第 1 层是通过对第 0 层图像进行卷积获得的，因此如果使用相对于与 $I_n(x,y)$ 尺寸大小相同的 $I_s(x,y)$ 的尺度参数进行卷积，则效果并不是我们想要的，应该使用其一半的尺度参数进行卷积才能得到我们想要的尺度。

综上，我们再对尺度进行一次总结。在 SIFT 算法中，尺度是指获取图像时，相对于原始图像 $I_s(x,y)$ 使用的高斯卷积的方差参数，但是这里原始图像 $I_s(x,y)$ 的尺寸大小不同，也会导致对尺度的描述不同。例如第 0 组第 0 层图像，它相对于尺寸跟它一样的原始图像 $I_s(x,y)$ 的尺度为 $\sigma_0=1.6$，即它可以通过对尺寸和它一样的原始图像 $I_s(x,y)$ 使用尺度参数为 $\sigma_0=1.6$ 的高斯函数卷积获得；而第 1 组第 0 层图像，它的尺度为 $2\sigma_0=3.2$，这里有三种不同的方式获得它，首先是对第 0 组图像中尺度为 $\sigma_{0,i}=2\sigma_0=3.2$ 的第 i 层图像降采样获得。其次可以先对尺寸是它两倍（尺寸与第 0 组图像的相同）的原始图像 $I_s(x,y)$ 使用尺度参数为 $\sigma_0=1.6$

的高斯函数卷积，再降采样获得与它相同尺寸的图像获得。也可以先对尺寸是它两倍（尺寸与第 0 组图像的相同）的原始图像 $I_s(x,y)$ 降采样，获得尺寸和它一样的原始图像 $I_s(x,y)$，再对该原始图像使用尺度参数为 $\sigma_0=1.6$ 的高斯函数卷积获得。由于这两种不同的获得方式，因此对金字塔中的图像尺度，可以有两种不同的描述方式，首先是相对于与第 0 组图像尺寸相同的原始图像的尺度 $\sigma_{o,s}$，我称它为金字塔级尺度，可以用它来描述金字塔中所有层的图像，且它们都有相同的基准原始图像；其次是相对于与每组自己尺寸相同的原始图像的尺度 σ_oct_n，每层有自己尺寸大小的基准原始图像。到现在为止，我们讨论的尺度，都是第一种尺度，即相对于与第 0 组图像尺寸相同的原始图像的尺度 $\sigma_{o,s}$，在后文，我们会用到 σ_oct_s。还是举例来说明，第 0 组第 0 层的 $\sigma_{0,0}$ 为 $\sigma_0=1.6$，$\sigma\_oct_0=\sigma_0=1.6$；第 0 组第 1 层的 $\sigma_{0,1}=k\sigma_{0,0}=k\sigma_0=1.6k$，$\sigma\_oct_1=k\sigma\_oct_0=k\sigma_0=1.6k$，第 0 组第 2 层的 $\sigma_{0,2}=k^2\sigma_{0,0}=k^2\sigma_0=1.6k^2$，$\sigma\_oct_2=k^2\sigma\_oct_0=k^2\sigma_0=1.6k^2$；第 0 组其他层依此类推，可以看到第 0 组的金字塔级尺度和组内尺度是相同的。第 1 组第 0 层的 $\sigma_{1,0}$ 为 $\sigma_{1,0}=2\sigma_{0,0}=2\sigma_0=3.2$，$\sigma\_oct_0=\sigma_0=1.6$；第 1 组第 1 层的 $\sigma_{1,1}=k\sigma_{1,0}=k\sigma_0=1.6k$，$\sigma\_oct_1=k\sigma\_oct_0=k\sigma_0=1.6k$，第 1 组其他层依此类推。因此**相邻两组的同一层尺度为 2 倍的关系是指**金字塔级的尺度 $\sigma_{o,s}$，而对于组内尺度 σ_oct_s 来说，每组相同层之间的 σ_oct_s 是相同的，且与第 0 组的金字塔级尺度 $\sigma_{0,s}$ 相同。

获得了每组第 0 层后，获得其他层的方法是对前一层图像进行高斯卷积，因此高斯卷积的相对尺度也应该根据组内尺度获得，因为金字塔级的尺度都是相对于与第 0 组图像尺寸相同的原始图像来说的尺度，而组内尺度才是与每组自己尺寸相同的原始图像来说的尺度。这样，第 1 组第 1 层相对尺度的获取方式为 $\sigma_{1,diff0}=\sqrt{(k\sigma\_oct_0)^2-(\sigma\_oct_0)^2}=\sqrt{(k\sigma_0)^2-(\sigma_0)}=\sigma_{0,diff0}$，其他层以此类推，因

此可得第一组每层的相对尺度与第 0 组相同。因此可以获得第 1 组的尺度空间表示（相对于与第 0 层尺寸相同的原始图像 $I_s(x,y)$ 来说）：

组数（Octave）	层数（Interval）	尺度空间表示	尺度参数 $\sigma_{o,s}$	相对于前一层的尺度参数 $\sigma_{1,diff}$
0	0	$L(x,y;2\sigma_0)$	σ_0	σ_0（相对于原始图像的尺度）
	1	$L(x,y;k\times 2\sigma_0)$	$k\sigma_0$	$\sqrt{(k\sigma\_oct_0)^2-\sigma\_oct_0^2}$
	2	$L(x,y;k^2\times 2\sigma_0)$	$k^2\sigma_0$	$\sqrt{(k^2\sigma\_oct_0)^2-(k\sigma\_oct_0)^2}$
	3	$L(x,y;k^3\times 2\sigma_0)$	$k^3\sigma_0$	$\sqrt{(k^3\sigma\_oct_0)^2-(k^2\sigma\_oct_0)^2}$
	4	$L(x,y;k^3\times 2\sigma_0)$	$k^4\sigma_0$	$\sqrt{(k^4\sigma\_oct_0)^2-(k^3\sigma\_oct_0)^2}$
	⋮	⋮	⋮	⋮
	s	$L(x,y;k^s\times 2\sigma_0)$	$k^s\sigma_0$	$\sqrt{(k^s\sigma\_oct_0)^2-(k^{s-1}\sigma\_oct_0)^2}$

高斯金字塔其他组图像的生成、

高斯金字塔中其他组的图像的生成方式与第二组相同，按照上文说法，其他层图像通过对前一层图像用同样的相对尺度的高斯函数进行卷积获得。其他层的尺度表示就不一一列出了。

k 的取值

为了在每组中检测 S（大写，与前文每组层数索引小写 s 相区分）个尺度的极值点，则 DOG 金字塔每组需 S+2 层图像，因为每相邻的三个 DOG 图像可以获取一个尺度的极值点，例如有 5 个连续的 DoG 图像，则第 1、2、3 层 DoG 图像可以获得一个尺度的极值点，第 2、3、4 层图像可以获得一个尺度的极值点，第 3、4、5 层图像可以获得一个尺度的极值点，最终可以获得 3 个尺度的极值点。而 DOG 金字塔由高斯金字塔相邻两层相减得到，则高斯金字塔每组需 S+2 层图

像，实际计算时 S 在 3 到 5 之间。按照前文对每组各层尺度空间的计算方法，首先计算第一组的尺度空间表示为$L(x,y;\sigma_0)$、$L(x,y;k\sigma_0)$、$L(x,y;k^2\sigma_0)$、...、$L(x,y;k^S\sigma_0)$、$L(x,y;k^{S+1}\sigma_0)$、$L(x,y;k^{S+2}\sigma_0)$。其对应的 DoG 图像为$D(x,y;\sigma_0)$、$D(x,y;k\sigma_0)$、$D(x,y;k^2\sigma_0)$、...、$D(x,y;k^S\sigma_0)$、$D(x,y;k^{S+1}\sigma_0)$。这样可以获得极值点的图像尺度参数分别为$k\sigma_0$、$k^2\sigma_0$、...、$k^S\sigma_0$。之后需要计算的第二组的尺度空间表示，有前文可以确定的是，第二组第一层的尺度空间参数应该是第一组第一层的两倍，因此它的尺度空间表示为$L(x,y;2\sigma_0)$。第二组其他层的尺度空间分别是$L(x,y;k\times2\sigma_0)$、$L(x,y;k^2\times2\sigma_0)$、...、$L(x,y;k^S\times2\sigma_0)$、$L(x,y;k^{S+1}\times2\sigma_0)$、$L(x,y;k^{S+2}\times2\sigma_0)$。通过相邻两层高斯金字塔相减可以获得 DoG 金字塔，其对应的 DoG 图像分别是$D(x,y;2\sigma_0)$、$D(x,y;k\times2\sigma_0)$、...、$D(x,y;k^S\times2\sigma_0)$、$D(x,y;k^{S+1}\times2\sigma_0)$。这样通过相邻三层 DoG 图像获得极值点的尺度参数分别为$k\times2\sigma_0$、...、$k^S\times2\sigma_0$。

现在来看第二组第一个极值点尺度参数和第一组最后一个极值点尺度参数分别为$k\times2\sigma_0$和$k^S\sigma_0$。我们已知的是各组内的尺度参数是连续的，现在我们希望这两个不同组间相邻极值点的尺度参数（金字塔级尺度）也是连续的，因此有如下关系：

$$k^{S+1}\sigma_0 = k\times2\sigma_0 \tag{4.14}$$

因此可以得到

$$k^S = 2 \tag{4.15}$$

最后可以得到

$$k = 2^{\frac{1}{S}} \tag{4.16}$$

除了第二组和第一组间相邻的尺度参数连续外，也要保证其他组间的尺度参数连续，具体求解方法与第二组和第一组间相邻的尺度参数的计算方法相同，结果也相同，因此整个高斯金字塔内的 k 值都是式(4.16)所求出的值。

也因为这样，所以高斯金字塔第 i 组第 0 层的图像（尺度参数为$2^i\sigma$），是

根据前一组倒数第三层（尺度参数为$k^S \times 2^{i-1}\sigma = 2^i\sigma$）降采样获得，可以看到使用式(4.16)计算获得的前一组倒数第三层的尺度参数与第 i 组第一层的尺度参数相同。这里可以看出来降采样没有对尺度参数产生影响。

构建尺度空间

其实前文已经把尺度空间构建出来了，现在只是把之前构建的尺度空间进行一次总结。

首先需要再强调的是，降采样时，金字塔上边一组图像的第一张图像（该组最底层的一张）是由前一组（金字塔下面一组）图像的倒数第三张隔点采样得到。

在高斯金字塔中，金字塔级尺度σ和 o、s 的关系如下:

$$\sigma(o,s) = \sigma_0 2^{o+\frac{s}{S}} \tag{4.17}$$

其中σ_0是基准层尺度，o 为 Octave 的索引，$o \in [0,\cdots,O-1]$，s 为组图像的层索引，$s \in [0,\cdots,S+2]$。S 为每组需要获得的极值点的图像数（一般为 3~5）。式(4.17)就是最后获得的高斯金字塔的尺度空间。另外还有组内尺度$\sigma_oct(\mathrm{s})$表示如下：

$$\sigma\_oct(\mathrm{s}) = \sigma_0 2^{\frac{s}{S}} \tag{4.18}$$

尺度空间生成了多少幅图像

我们知道$D(x,y;\sigma)$是我们最终构建出来的用来寻找关键点的高斯差分图像，而关键点的寻找需要查找的是空间局部极小值，即在某一层上查找局部极值点的时候需要用到上一层与下一层的高斯差分图像，所以如果我们需要查找 S 层的关键点，需要 S+2 层高斯差分图像，然后查找其中的第 2 层到第 S+1 层。而每一个高斯差分图像$G(x,y;\sigma)$都需要两幅尺度空间的图像$G(x,y;k\sigma)$与$G(x,y;\sigma)$进行差分生成，这里假设 S =3，则我们需要的高斯差分图像有 S+2 = 5 层，需要的高斯图像有 S+3=6 层。根据式(4.8)的计算，我们可以知道整个尺度空间有 n 组，因此总共需要的$n(S+3)$层尺度图像。

第五章 关键点搜索

5.1 极值点搜索

与前文介绍的 LoG 算子搜索斑点的过程相同。关键点的搜索是通过同一组内各 DoG 相邻层之间比较完成的。为了寻找尺度空间的极值点，每一个采样点要和它所有的相邻点进行比较，看其是否比它的图像域和尺度域的相邻点大或小。对于其中的任意一个检测点都要和它同尺度的 8 个相邻点和上下相邻尺度对应的9×2个点共 26 个点比较，以确保在尺度空间和二维图像位置空间都检测到极值点。也就是，比较是在一个3×3的立方体内进行。

搜索过程从每组的第二层开始，以第二层为当前层，对第二层的 DoG 图像中的每个点取一个3×3的立方体，立方体上下层为第一层与第三层。这样，搜索得到的极值点既有位置坐标（DoG 的图像坐标），又有空间尺度坐标（层坐标）。当第二层搜索完成后，再以第三层作为当前层，其过程与第二层的搜索类似。当$S=3$时，每组里面要搜索 3 层。

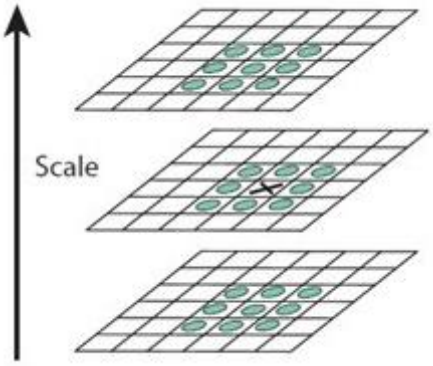


图 5-1 极值点搜索策略

5.2 子像元插值

前文的极值点的搜索是在离散空间中进行的，检测到的极值点并不是真正意义上的极值点。下图显示了一维信号离散空间得到的极值点与连续空间的极值点之间的差别。利用已知的离散空间点插值到连续空间极值点的方法叫子像元插值。

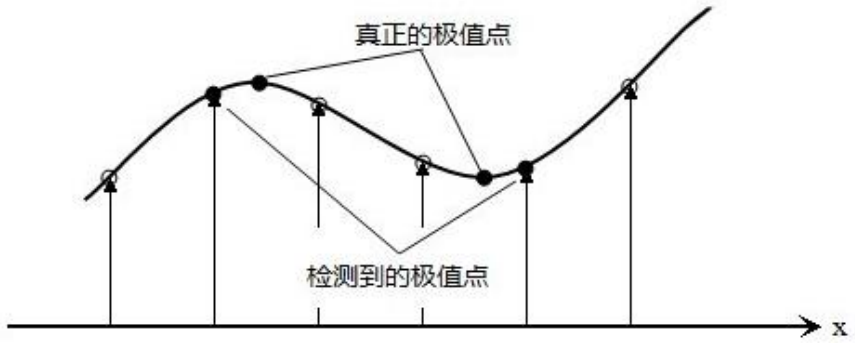


图 5-2 离散空间的极值点与实际极值点

首先我们来看一个一维函数插值的例子。我们已经 $f(x)$ 上几个点的函数值 $f(-1)=-1$， $f(0)=6$， $f(1)=5$，求 $f(x)$ 在 $[-1,1]$ 上的最大值。

如果我们只考虑离散的情况，那么只用简单比较一下，便知最大值为 $f(0)=6$，下面我们用子像元插值法来考虑连续区间的上情况：

利用泰勒级数，可以将 $f(x)$ 在 $f(0)=6$ 附近展开为：

$$f(x)\approx f(0)+f'(0)x+\frac{f''(0)}{2}x^2 \tag{5.1}$$

另外我们知道 $f(x)$ 在 x 的导数写成离散的形式为 $f'(\mathrm{x})=\frac{\mathrm{f(x+1)-f(x)}}{1}$，二阶导数写成离散形式为 $f''(x)=f(x+1)+f(x-1)-2f(x)$。所以我们可以计算出 $f(x)\approx 6+2x+\frac{-6}{2}x^2=6+2x-3x^2$。求取 $f(x)$ 极大值以及极大值所在的位置

$$f'(\mathrm{x})=2-6\,\mathrm{x}=0,\tilde{\mathrm{x}}=\frac{1}{3}$$

$$f(\tilde{\mathrm{x}})=6+2\times\frac{1}{3}-3\times\left(\frac{1}{3}\right)^2=6\frac{1}{3}$$

现在回到我们 SIFT 点检测中来，我们要考虑的是一个三维问题，假设我们在尺度为 σ 的尺度图像 $D(x,y)$ 上检测到了一个局部极值点，空间位置为 (x,y,σ)，由上面的分析我们知道，它只是一个离散情况下的极值点，连续情况下，极值点可能落在了 (x,y,σ) 的附近，设其偏离了 (x,y,σ) 的坐标为 $(\Delta x,\Delta y,\Delta\sigma)$。则对 $D(\Delta x,\Delta y,\Delta\sigma)$ 可以表示为在点 (x,y,σ) 处的泰勒展开：

$$D(\Delta x,\Delta y,\Delta\sigma)=D(\mathrm{x},\mathrm{y},\sigma)+\left[\begin{array}{ccc}\frac{\partial D}{\partial x} & \frac{\partial D}{\partial y} & \frac{\partial D}{\partial z}\end{array}\right]\Bigg|_{(\mathrm{x},\mathrm{y},\sigma)}\left[\begin{array}{c}\Delta x\\ \Delta y\\ \Delta\sigma\end{array}\right]$$
$$+\frac{1}{2}\left[\begin{array}{ccc}\Delta x & \Delta y & \Delta\sigma\end{array}\right]\left[\begin{array}{ccc}\frac{\partial^2 D}{\partial x^2} & \frac{\partial^2\Omega}{\partial x\partial y} & \frac{\partial^2\Omega}{\partial x\partial\sigma}\\ \frac{\partial^2\Omega}{\partial y\partial x} & \frac{\partial^2 D}{\partial y^2} & \frac{\partial^2\Omega}{\partial y\partial\sigma}\\ \frac{\partial^2\Omega}{\partial\sigma\partial x} & \frac{\partial^2\Omega}{\partial\sigma\partial y} & \frac{\partial^2 D}{\partial\sigma^2}\end{array}\right]\Bigg|_{(\mathrm{x},\mathrm{y},\sigma)}\left[\begin{array}{c}\Delta x\\ \Delta y\\ \Delta\sigma\end{array}\right] \tag{5.2}$$

将上式写成矢量形式，令 $\boldsymbol{t}=\left[\begin{array}{c}x\\ y\\ \sigma\end{array}\right]$ ，则式(5.2)可以表示成如下形式：

$$D(\Delta\boldsymbol{t})=D(\boldsymbol{t})+\nabla D(\boldsymbol{t})^{\mathrm{T}}\Delta\boldsymbol{t}+\frac{1}{2}\Delta\boldsymbol{t}^{\mathrm{T}}\frac{\partial^2 D}{\partial\boldsymbol{t}^2}\Delta\boldsymbol{t} \tag{5.3}$$

其中 $\nabla D(\boldsymbol{t})$ 是 DoG 图像在 $\boldsymbol{t}=\left[\begin{array}{ccc}x & y & \sigma\end{array}\right]^{\mathrm{T}}$ 处的梯度，$\frac{\partial^2 D}{\partial\boldsymbol{t}^2}$ 为 DoG 图像在 $\boldsymbol{t}=\left[\begin{array}{ccc}x & y & \sigma\end{array}\right]^{\mathrm{T}}$ 处的 Hessian 矩阵。

对式(5.3)求导，得到

$$D'(\Delta\boldsymbol{t})=\nabla D(\boldsymbol{t})+\frac{\partial^2 D}{\partial\boldsymbol{t}^2}\Delta\boldsymbol{t} \tag{5.4}$$

令其为零，可得到最终的结果

$$\Delta\boldsymbol{t}=-\left[\frac{\partial^2 D}{\partial\boldsymbol{t}^2}\right]^{-1}\nabla D(\boldsymbol{t}) \tag{5.5}$$

极值点的极值为

$$D(\boldsymbol{t}+\Delta\boldsymbol{t})=D(\boldsymbol{t})+\frac{1}{2}\frac{\partial D^T}{\partial\boldsymbol{t}}\Delta\boldsymbol{t} \tag{5.6}$$

其中，$\Delta\boldsymbol{t}=\left[\begin{array}{ccc}\Delta x & \Delta y & \Delta\sigma\end{array}\right]^{\mathrm{T}}$ 代表相对插值中心的偏移量，当它在任一维度上的偏移量大于 0.5 时（即 Δx 或 Δy 或 $\Delta\sigma$ ），意味着插值中心已经偏移到它的邻近点上，所以必须改变当前关键点的位置。同时在新的位置上反复插值直到收敛；也有可能超出所设定的迭代次数或者超出图像边界的范围，此时这样的点应该删除，在

Lowe 中进行了 5 次迭代。另外，过小的点易受噪声的干扰而变得不稳定，所以将$\left|D(\boldsymbol{t}+\Delta\boldsymbol{t})\right|$小于某个经验值（Lowe 论文中使用 0.03，Rob Hess 等人实现时使用$\frac{0.04}{S}$，假设极值点图像的灰度值是在 0 到 1.0 之间）的极值点删除。同时，在此过程中获取关键点的精确位置（原位置加上拟合的偏移量）以及尺度（$\sigma(x,y)$）。

3.3 消除边缘响

一个定义不好的高斯差分算子的极值在横跨边缘的地方有较大的主曲率，而在垂直边缘的方向有较小的主曲率。

DOG 算子会产生较强的边缘响应，需要剔除不稳定的边缘响应点。获取关键点处的 Hessian 矩阵，主曲率通过一个2×2的 Hessian 矩阵 H 求出：

$$H=\begin{bmatrix}D_{xx} & D_{xy}\\ D_{yx} & D_{yy}\end{bmatrix} \tag{5.7}$$

H 的特征值α和β代表 x 和 y 方向的梯度，

$$\begin{aligned}Tr(H)&=D_{xx}+D_{yy}=\alpha+\beta\\ Det(H)&=D_{xx}D_{yy}-\left(D_{xy}\right)^2=\alpha\beta\end{aligned} \tag{5.8}$$

其中$Tr(H)$表示矩阵 H 对角线元素之和，$Det(H)$表示矩阵 H 的行列式。假设α是较大的特征值，而β是较小的特征值，令$\alpha=r\beta$，则

$$\frac{Tr(H)}{Det(H)}=\frac{(\alpha+\beta)}{\alpha\beta}=\frac{(r\beta+\beta)}{r\beta^2}=\frac{(r+1)^2}{r} \tag{5.9}$$

导数由采样点相邻差估计得到,在下一节中说明。

D 的主曲率和 H 的特征值成正比，令α为最大特征值，β为最小的特征值，则公式的值在两个特征值相等时最小，随着的增大而增大。值越大，说明两个特征值的比值越大，即在某一个方向的梯度值越大，而在另一个方向的梯度值越小，而边缘恰恰就是这种情况。所以为了剔除边缘响应点，需要让该比值小于一定的阈值，因此，为了检测主曲率是否在某域值 r 下，只需检测

$$\frac{Tr(H)}{Det(H)}<\frac{(r+1)^2}{r} \tag{5.10}$$

是否成立，若成立则保留关键点，反之剔除。在 Lowe 的文章中，取 $r=10$。

3.4 有限差分法求导

有限差分法以变量离散取值后对应的函数值来近似微分方程中独立变量的连续取值。在有限差分方法中，我们放弃了微分方程中独立变量可以取连续值的特征，而关注独立变量离散取值后对应的函数值。但是从原则上说，这种方法仍然可以达到任意满意的计算精度。因为方程的连续数值解可以通过减小独立变量离散取值的间格，或者通过离散点上的函数值插值计算来近似得到。这种方法是随着计算机的诞生和应用而发展起来的。其计算格式和程序的设计都比较直观和简单，因而，它在计算数学中使用广泛。

有限差分法的具体操作分为两个部分：

1. 用差分代替微分方程中的微分，将连续变化的变量离散化，从而得到差分方程组的数学形式；

2. 求解差分方程组。

一个函数在 x 点上的一阶和二阶微商，可以近似地用它所临近的两点上的函数值的差分来表示。如对一个单变量函数 $f(x)$，x 为定义在区间 $[a,b]$ 上的连续变量，以步长 $h=\Delta x$ 将区间 $[a,b]$ 离散化，我们会得到一系列节点，

$$x_1=a, x_2=x_1+h=a+h, \cdots, x_{n+1}=x_n+1=b$$

然后求出 $f(x)$ 在这些点上的近似值。显然步长 h 越小，近似解的精度就越好。

与节点 x_i 相邻的节点有 x_i-h 和 x_i+h，所以在节点 x_i 处可构造如下形式的差值：

$f(x_i+h)-f(x_i)$ 节点的一阶向前差分

$f(x_i)-f(x_i-\mathrm{h})$ 节点的一阶向后差分

$f(x_i+h)-f(x_i-\mathrm{h})$ 节点的一阶中心差分

这里使用中心差分利用泰勒展开式求解本节所用的偏导数，现做如下推到：

函数 $f(x)$ 在 x_i 处的泰勒展开式为：

$$f(x)=f(x_i)+f'(x_i)(x-x_i)+\frac{f''(x_i)}{2!}(x-x_i)^2+\cdots+\frac{f^{(n)}(x_i)}{n!}(x-x_i)^n \quad (5.11)$$

则

$$f(x_i-h)=f(x_i)+f'(x_i)((x_i-h)-x_i)+\frac{f''(x_i)}{2!}((x_i-h)-x_i)^2+\cdots+\frac{f^{(n)}(x_i)}{n!}((x_i-h)-x_i)^n \quad (5.12)$$

$$f(x_i+h)=f(x_i)+f'(x_i)((x_i+h)-x_i)+\frac{f''(x_i)}{2!}((x_i+h)-x_i)^2+\cdots+\frac{f^{(n)}(x_i)}{n!}((x_i+h)-x_i)^n \quad (5.13)$$

忽略 h 平方之后的项，联立式(5.12)、式(5.13)解方程组得

$$f'(x_i)=\left(\frac{\partial f}{\partial x}\right)_{x_i}\approx\frac{f(x_i+h)-f(x_i-h)}{2h} \quad (5.14)$$

$$f'(x_i)=\left(\frac{\partial^2 f}{\partial x^2}\right)_{x_i}\approx\frac{f(x_i+h)+f(x_i-h)-2f(x_i)}{h^2} \quad (5.15)$$

二元函数的泰勒展开式如下：

$$f(\mathrm{x}+\Delta\mathrm{x},\mathrm{y}+\Delta\mathrm{y})=\mathrm{f}(\mathrm{x},\mathrm{y})+\Delta\mathrm{x}\frac{\partial f(\mathrm{x},\mathrm{y})}{\partial x}+\Delta y\frac{\partial f(\mathrm{x},\mathrm{y})}{\partial y}$$
$$+\frac{1}{2!}\left[(\Delta\mathrm{x})^2\frac{\partial^2 f(\mathrm{x},\mathrm{y})}{\partial x^2}+2\Delta\mathrm{x}\,\Delta y\frac{\partial^2 f(\mathrm{x},\mathrm{y})}{\partial x\partial y}+(\Delta y)^2\frac{\partial^2 f(\mathrm{x},\mathrm{y})}{\partial y^2}\right]$$
$$+\frac{1}{3!}\left[(\Delta\mathrm{x})^3\frac{\partial^3 f(\mathrm{x},\mathrm{y})}{\partial x^3}+3(\Delta\mathrm{x})^2\Delta y\frac{\partial^3 f(\mathrm{x},\mathrm{y})}{\partial x^2\partial y}+3\Delta\mathrm{x}(\Delta y)^2\frac{\partial^3 f(\mathrm{x},\mathrm{y})}{\partial x\partial y^2}+(\Delta y)^3\frac{\partial^3 f(\mathrm{x},\mathrm{y})}{\partial y^3}\right]+\cdots$$

将 $f(x_i+h,y_i+h)$、$f(x_i+h,y_i-h)$、$f(x_i-h,y_i+h)$ 和 $f(x_i-h,y_i-h)$ 展开后忽略此要项联立解方程得二维混合偏导如下：

$$\frac{\partial^2 f(\mathrm{x}_i,\mathrm{y}_i)}{\partial x\partial y}\approx\frac{1}{4h^2}[f(x_i+h,y_i+h)+f(x_i-h,y_i-h) -f(x_i+h,y_i-h)-f(x_i-h,y_i+h)] \quad (5.16)$$

综上，推导了 3.3、3.2 节遇到的所有导数计算。同理，利用多元泰勒展开式，可得任意偏导的近似差分表示。

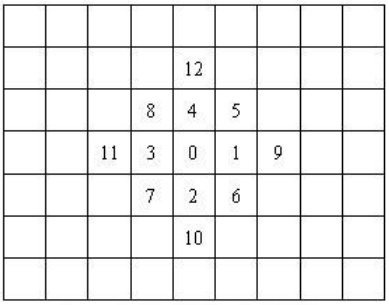


图 5-3 图像中像素 0 与其邻域

在图像处理中，取 h=1，在图 3-3 所示的图像中，将像素 0 的基本中点导数公式整理如下：

$$\left(\frac{\partial f}{\partial x}\right)_0 = \frac{f_1 - f_3}{2h}$$

$$\left(\frac{\partial f}{\partial y}\right)_0 = \frac{f_2 - f_4}{2h}$$

$$\left(\frac{\partial^2 f}{\partial x^2}\right)_0 = \frac{f_1 + f_3 - 2f_0}{h^2}$$

$$\left(\frac{\partial^2 f}{\partial y^2}\right)_0 = \frac{f_2 + f_4 - 2f_0}{h^2}$$

$$\left(\frac{\partial^2 f}{\partial x \partial y}\right)_0 = \frac{(f_8 + f_6) - (f_5 + f_7)}{4h^2}$$

$$\left(\frac{\partial^4 f}{\partial x^4}\right)_0 = \frac{6f_0 - 4(f_1 + f_3) + (f_9 + f_{11})}{h^4}$$

$$\left(\frac{\partial^4 f}{\partial y^4}\right)_0 = \frac{6f_0 - 4(f_2 + f_4) + (f_{10} + f_{12})}{h^4}$$

$$\left(\frac{\partial^4 f}{\partial x^2 \partial y^2}\right)_0 = \frac{4f_0 - 2(f_1 + f_2 + f_3 + f_4) + (f_5 + f_6 + f_7 + f_8)}{h^4}$$

3.5 三阶矩阵求逆公式

高阶矩阵的求逆算法主要有归一法和消元法两种，现将三阶矩阵求逆公式总结如下：

若矩阵

$$A = \begin{pmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{pmatrix}$$

可逆，即

$$|A| = a_{00}a_{11}a_{22} + a_{01}a_{12}a_{20} + a_{02}a_{10}a_{21} - a_{00}a_{12}a_{21} - a_{01}a_{10}a_{22} - a_{02}a_{11}a_{20} \neq 0$$

时

$$A^{-1} = \frac{1}{|A|}\begin{pmatrix} a_{11}a_{22} - a_{21}a_{12} & -(a_{01}a_{22} - a_{21}a_{02}) & a_{01}a_{12} - a_{02}a_{11} \\ a_{12}a_{20} - a_{22}a_{10} & -(a_{02}a_{20} - a_{22}a_{00}) & a_{02}a_{10} - a_{00}a_{12} \\ a_{10}a_{21} - a_{20}a_{11} & -(a_{00}a_{21} - a_{20}a_{01}) & a_{00}a_{11} - a_{01}a_{10} \end{pmatrix} \tag{5.17}$$

第六章 关键点方向赋值

为了实现图像旋转不变性，需要根据检测到的关键点局部图像结构为关键点方向赋值。我们使用图像的梯度直方图法求关键点局部结构的稳定方向。

6.1 梯度方向和幅值

对于在 DOG 金字塔中检测出的关键点点，采集其所在高斯金字塔图像 3σ 邻域窗口内像素的梯度和方向分布特征。梯度的模值和方向如下：

$$\begin{aligned} m(x,y) &= \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2} \\ \theta(x,y) &= \tan^{-1}\left(\frac{(L(x,y+1) - L(x,y-1))}{(L(x+1,y) - L(x-1,y))}\right) \end{aligned} \tag{6.1}$$

L 为关键点所在的尺度空间值，由于邻域内的像素梯度幅值对圆心处的关键点的贡献是不同的，因此按 Low 的建议，梯度的模值 $m(x,y)$ 按 $\sigma = 1.5\sigma\_oct$ 的高斯分布加成，按尺度采样的 3σ 原则，邻域窗口半径为 $3\times 1.5\sigma\_oct$ 。

在完成关键点的梯度计算后，使用直方图统计**邻域**内像素的梯度和方向，其中梯度需要加权计算。梯度直方图将 0~360 度的方向范围分为 36 个柱（bins），其中每柱 10 度。如图 4.1 所示，直方图的峰值方向代表了关键点的主方向(为简化，图中只画了八个方向的直方图)。

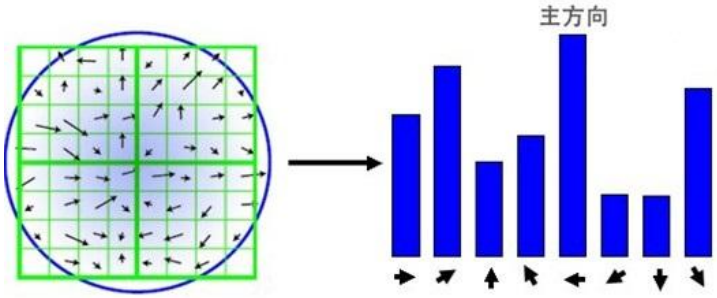


图 6-1 关键点方向直方图

直方图峰值代表该关键点处邻域内图像梯度的方向，以直方图中最大值作为该关键点的主方向。在梯度方向直方图中，当存在另一个大于等于主峰值 80%能量的峰值时，则将这个方向认为是该关键点的辅方向。所以一个关键点可能检测得到多个方向，这可以增强匹配的鲁棒性。Lowe 的论文指出大概有 15%关键点具有多方向，但这些点对匹配的稳定性至为关键。

实际编程实现中，就是把该关键点复制成多份关键点，并将方向值分别赋给这些复制后的关键点，并且，离散的梯度方向直方图要进行插值拟合处理，来求得更精确的方向角度值，检测结果如图 6-2 所示。

在计算直方图时，每个加入直方图的采样点都使用圆形高斯函数函数进行了加权处理，也就是进行高斯平滑，减少突变的影响。这主要是因为 SIFT 算法只考虑了尺度和旋转不变形，没有考虑仿射不变性。通过高斯平滑，可以使关键点附近的梯度幅值有较大权重，从而部分弥补没考虑仿射不变形产生的关键点不稳定。

通常离散的梯度直方图要进行插值拟合处理，以求取更精确的方向角度值。（这和《关键点搜索》中插值的思路是一样的）。或 $h(\mathrm{i})$ 为第 i 柱所代表的梯度，则使用下式对其进行平滑：

$$H(\mathrm{i})=\frac{h(\mathrm{i}-2)+\mathrm{h}(\mathrm{i}+2)}{16}+\frac{4\times(\mathrm{h}(\mathrm{i}-1)+\mathrm{h}(\mathrm{i}+1))}{16}+\frac{6\times h(\mathrm{i})}{16} \tag{6.2}$$

其中，h 和 H 分别表示平滑前和平滑后的直方图。由于角度是循环的，即 $0°=360°$，如果计算过程中出现 $h(\mathrm{j})$，j 超出了 $(0,1,\cdots 15)$ 的范围，那么可以通过圆周循环的方法找到它所对应的、在 $0°\sim 360°$ 之间的值，如 $h(-1)=h(15)$。

另外，我们通过梯度直方图获得的关键点的主方向只是一个角度范围，例如

如果$\mathrm{i}=0$，则主方向为$0°\sim 9°$，要获得更精确的方向角度值，还需要对离散的梯度直方图进行插值拟合处理，拟合公式如下：

$$B=\mathrm{i}+\frac{H(\mathrm{i}-1)-H(\mathrm{i}+1)}{2\times(H(\mathrm{i}-1)+H(\mathrm{i}+1)-2\times H(\mathrm{i}))} \tag{6.3}$$

$$\theta=360-10\times B \tag{6.4}$$

其中，$\mathrm{i}=0,\cdots 15$。H 为式(6.2)得到的直方图，角度θ的单位是度。同样地，式(6.3)和式(6.4)也存在着式(6.2)的角度不在定义域范围内的问题，处理的方法还是利用角度的圆周循环。

获得图像关键点主方向后，每个关键点有三个信息(x,y,σ,θ)：位置、尺度、方向。由此我们可以确定一个 SIFT 特征区域。通常使用一个带箭头的圆或直接使用箭头表示 SIFT 区域的三个值：中心表示关键点位置，半径表示关键点尺度（$r=2.5\sigma$）,箭头表示主方向。具有多个方向的关键点可以复制成多份，然后将方向值分别赋给复制后的关键点。如下图：

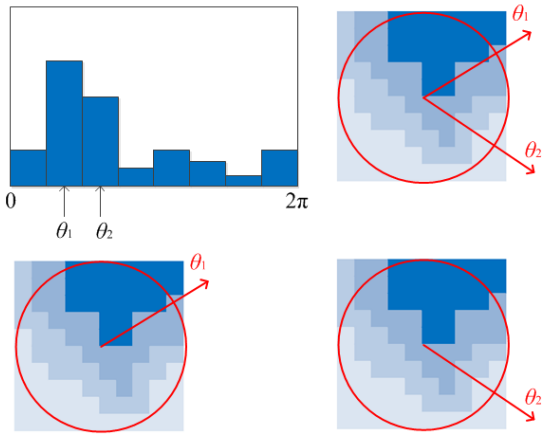


图 6-2 SIFT 特征区域

第七章 关键点特征描述

我们现在已经为关键点即 SIFT 特征点赋了值，包含位置、尺度和方向的信息。接下来就是为每个关键点建立一个描述符，用一组向量将这个关键点描述出来，使其不随各种变化而改变，比如光照变化、视角变化等等。这个描述子不但包括关键点，也包含关键点周围对其有贡献的像素点，并且描述符应该有较高的独特性，以便于提高关键点正确匹配的概率，用来作为目标匹配的依据（所以描述子应该有较高的独特性，以保证匹配率）。

SIFT 描述子是关键点邻域高斯图像梯度统计结果的一种表示。通过对关键点周围图像区域分块，计算块内梯度直方图，生成具有独特性的向量，这个向量是该区域图像信息的一种抽象，具有唯一性。

Lowe 建议描述子使用在关键点尺度空间内 4×4 个窗口中计算的 8 个方向的梯度信息，共 $4\times4\times8=128$ 维向量表征，每个窗口取 4×4 个像素进行计算。表示步骤如下：

1. 确定计算描述子所需的图像区域

特征描述子与关键点所在的尺度有关，因此，对梯度的求取应在关键点对应的高斯图像上进行。将关键点附近的邻域划分为 $d\times d$（Lowe 建议 $d=4$）个子区域，每个子区域做为一个种子点，每个种子点有 8 个方向。每个子区域的大小与关键点方向分配时相同，即每个区域有个 $3\sigma\_oct$ 子像素，为每个子区域分配边长为 $3\sigma\_oct$ 的矩形区域进行采样（每个子像素实际用边长为 $3\sigma\_oct$ 的矩形区域即可包含，但由于 $3\sigma\_oct\le6\sigma_0$ 不大，为了简化计算取其边长为 $3\sigma\_oct$，并且采样点宜多不宜少）。考虑到实际计算时，需要采用双线性插值，所需图像窗口边长为 $3\sigma\_oct\times(d+1)$。在考虑到旋转因素（方便下一步将坐标轴旋转到关键点的方向），如下图 5-1 所示，实际计算所需的图像区域半径为：

$$radius=\frac{3\sigma\_oct\times\sqrt{2}\times(d+1)}{2} \tag{7.1}$$

计算结果四舍五入取整数。

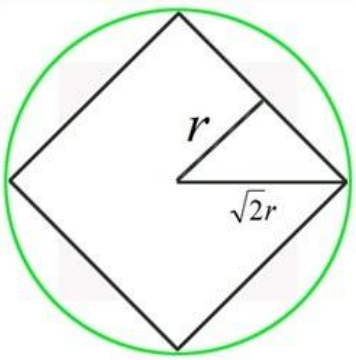


图 7-1 旋转引起的邻域半径变化

2. 将**坐标轴旋转为关键点的方向，则在旋转后的坐标系中，所有图像中的关键点方向都变为 0 度，这样能够确保关键点描述子的旋转不变性**，如 7-2 所示

旋转后邻域内采样点的新坐标为：

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{7.2}$$

其中 $x, y \in [-radius, radius]$。

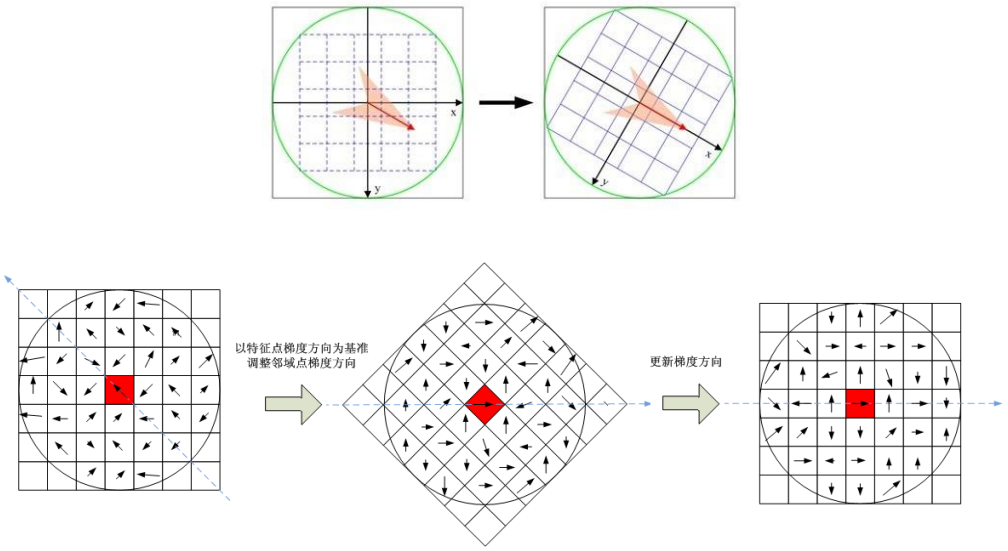


图 7-2 坐标系旋转

3. 计算采样区域梯度直方图，将邻域内的采样点分配到对应的子区域内，将子区域内的梯度值分配到 8 个方向上，计算其权值。

旋转后的采样点坐标在半径为 *radius* 的圆内被分配到 $d \times d$ 的子区域，计算影响子区域的采样点的梯度和方向，分配到 8 个方向上。

旋转后的采样点(x', y')落在子区域的下标为

$$\begin{pmatrix} x'' \\ y'' \end{pmatrix} = \frac{1}{3\sigma\_oct} \begin{pmatrix} x' \\ y' \end{pmatrix} + \frac{d}{2} \tag{7.3}$$

Lowe 建议子区域的像素的梯度大小按$\sigma = 0.5d$的高斯加权计算，即

$$w = m(a+x, b+y) * e^{-\frac{(x')^2+(y')^2}{2\times(0.5d)^2}} \tag{7.4}$$

其中 *a*，*b* 为关键点在高斯金字塔图像中的位置坐标。

4. 插值计算每个种子点八个方向的梯度。

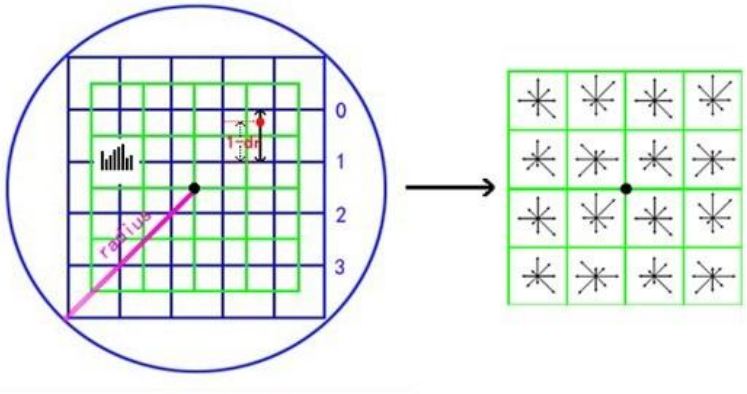


图 7-3 描述子梯度直方图

如图 7-3 所示，将由式(7.3)所得采样点在子区域中的下标（图中蓝色窗口内红色点）线性插值，计算其对每个种子点的贡献。如图中的红色点，落在第 0 行和第 1 行之间，对这两行都有贡献。对第 0 行第 3 列种子点的贡献因子为 *dr*，对第 1 行第 3 列的贡献因子为 1-*dr*，同理，对邻近两列的贡献因子为 *dc* 和 1-*dc*，对邻近两个方向的贡献因子为 *do* 和 1-*do*。则最终累加在每个方向上的梯度大小为：

$$weight = w \times dr^k \times (1-dr)^{1-k} \times dc^m \times (1-dc)^{1-m} \times do^n \times (1-do)^{1-n} \tag{7.5}$$

其中 *k*，*m*，*n* 为 0 或 1

5.统计的$4\times4\times8=128$个梯度信息即为该关键点的特征向量。特征向量形成后，为了去除光照变化的影响，需要对它们进行归一化处理，对于图像灰度值整体漂移，图像各点的梯度是邻域像素相减得到，所以也能去除。得到的描述子向量为$H=(h_1, h_2, \cdots, h_{128})$，归一化后的特征向量为$L=(l_1, l_2, \cdots, l_{128})$则

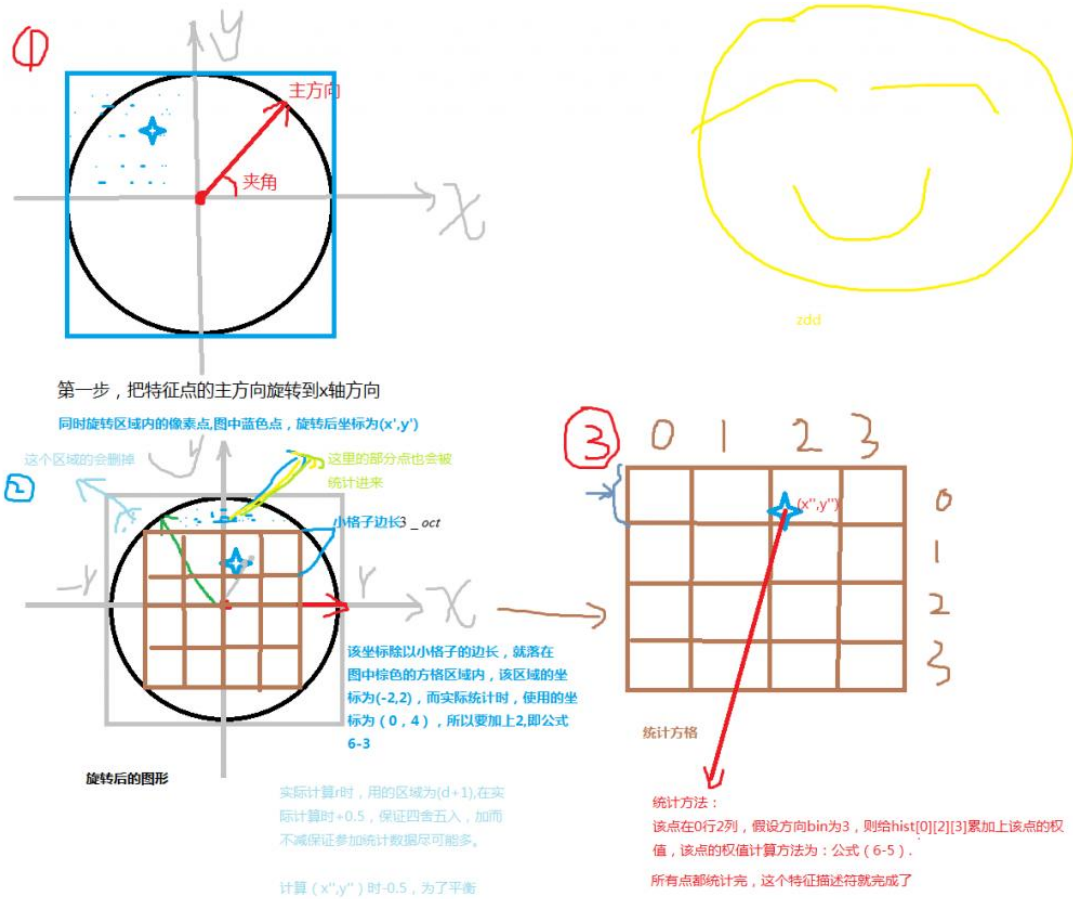
$$l_i = \frac{h_i}{\sqrt{\sum_{j=1}^{128} h_j}} \tag{7.6}$$

6.描述子向量门限。非线性光照，相机饱和度变化对造成某些方向的梯度值过大，而对方向的影响微弱。因此设置门限值（向量归一化后，一般取 0.2）断较大的梯度值。然后，再进行一次归一化处理，提高特征的鉴别性。

7.按关键点的尺度对特征描述向量进行排序。

至此，SIFT 特征描述向量生成。

描述向量这块不好理解，我画了个草图，供参考：



参考资料

https://blog.csdn.net/xiaowei_cqu/article/details/8069548；

https://zhuanlan.zhihu.com/p/36382429?utm_source=wechat_session&utm_medium=social&utm_oi=78907617312768；

http://www.cnblogs.com/starfire86/p/5735061.html；

http://www.cnblogs.com/ronny/p/4028776.html#3962356；

https://blog.csdn.net/zddblog/article/details/7521424；

https://blog.csdn.net/csuyhb/article/details/79836439；

https://blog.csdn.net/zhaocj/article/details/42124473；

https://wenku.baidu.com/view/d7edd2464b73f242336c5ffa.html；

有限差分法：

https://blog.csdn.net/zddblog/article/details/7521424；

LoG 算子和 DoG：

https://www.cnblogs.com/ronny/p/3895883.html；

https://blog.csdn.net/touch_dream/article/details/62237018；

https://blog.csdn.net/u014485485/article/details/78364573

高斯函数：

https://www.cnblogs.com/jermmyhsu/p/8251013.html；

https://blog.csdn.net/farmwang/article/details/78699926；

https://baike.baidu.com/item/%E4%BA%8C%E7%BB%B4%E6%AD%A3%E6%80%81%E5%88%86%E5%B8%83/2951835?fr=aladdin；

卷积：

https://blog.csdn.net/einstellung/article/details/77412903；

https://blog.csdn.net/liguan843607713/article/details/42215965；

https://blog.csdn.net/moonlightran/article/details/24374701；