雷达影像的图像处理技术

谢磊　博士 副研究员

国家水运安全工程技术研究中心

2021-03-01

联系方式

- 办公室：航海楼707
- 电话：18986175100
- QQ：812089913
- QQ群：雷达影像的图像处理技术2021，369368709
- 考试形式：大作业，采用图像处理或是神经网络的相关算法，对受到干扰的雷达图像进行基本的处理，并分析处理的效果。
- Matlab软件安装包

作业要求

- 作业以雷达影像为处理对象，要求学生实现多种图像处理算法，并对处理效果进行分析，以训练学生的matlab编程和算法分析能力。其中，课堂上讲解过的算法包括：图像读取和显示、颜色反转、直方图均衡化、噪声添加和滤波、图像边缘提取、傅里叶变化、人工神经网络等。作业应采用指定的模板，结合课堂教学内容，使用给定的雷达图像作为处理对象，进行算法处理和分析。需要应用的具体算法包括：图像读取和显示、颜色反转、添加噪声和滤波、图像边缘提取、傅里叶变化、人工神经网络（选做）。

- 作业应结合课堂教学内容，以给定的雷达图像为处理对象，进行算法处理和分析。在作业评分上鼓励学生大胆尝试多种图像处理算法，特别是傅里叶变化、人工神经网络等具有一定难度的算法。

第一章 图像处理技术简介（一）

一、数字图像基础

1、位图与矢量图

- 位图：由象素组成。这些由不同颜色的象素组成的画面，即为图像。图像是自然景物的直接反映。如：照片。摄影等。

- 矢量图：以数学的方式，对各种形状进行记录。这些由不同形状组成的画面，即为图形。图形是我们按照自己的理解表述出来的形状。如：一条线，一个圆，一个卡通人物等。

1、位图与矢量图

- 位图：由于位图就是由象素阵列的排列来实现其显示效果的，其最小单位为象素，因此缩放会失真。每个象素都有自己的颜色信息，在对位图图像进行编辑操作的时候，可操作的对象是每个象素，我们可以改变图像的色相、饱和度、亮度，从而改变图像的显示效果。通过编辑可以改变位图中任何区域的色彩显示效果；但是相应的，要实现的效果越复杂，需要的象素数越多，图像文件的大小和体积越大。

1、位图与矢量图

- 矢量图：矢量图可以通过多个对象的组合生成，对其中的每一个对象的纪录方式，都是以数学函数来实现的。也就是说，矢量图实际上并不是象位图那样纪录画面上每一点的信息，而是纪录了元素形状及颜色的算法。当你打开一张矢量图的时候，软件对图形上对应的函数进行运算，将运算结果（图形的形状和颜色）显示给你看。无论显示画面是大还是小，画面上的对象对应的算法是不变的，所以，即使对画面进行倍数相当大的缩放，其显示效果仍然相同，不会失真。矢量图轮廓的形状更容易修改和控制，但是对于单独的对象，色彩上变化的实现不如位图来的方便直接。另外，支持矢量格式的应用程序也远远没有支持位图的多，很多矢量图形都需要专门设计的程序才能打开、浏览和编辑。

2、亮度

- 因为数字图像作为离散的亮点集显示，所以眼睛对不同亮度之间的鉴别能力在图像处理结果中是要考虑的重要方面。
- 人的视觉系统能够适应的光强度级别范围是很宽的，从夜视阈值到强闪光约有10^{10}量级。
- 实验数据指出，主观亮度（即由人的视觉系统感觉到的亮度）是进入眼睛的光强度的对数函数。
- 所谓光亮度，就是人对光的心理感受程度。作为物理量，一方面，它与光强度对应；另一方面，它与光波长和眼的感受能力也有关系。

3、颜色

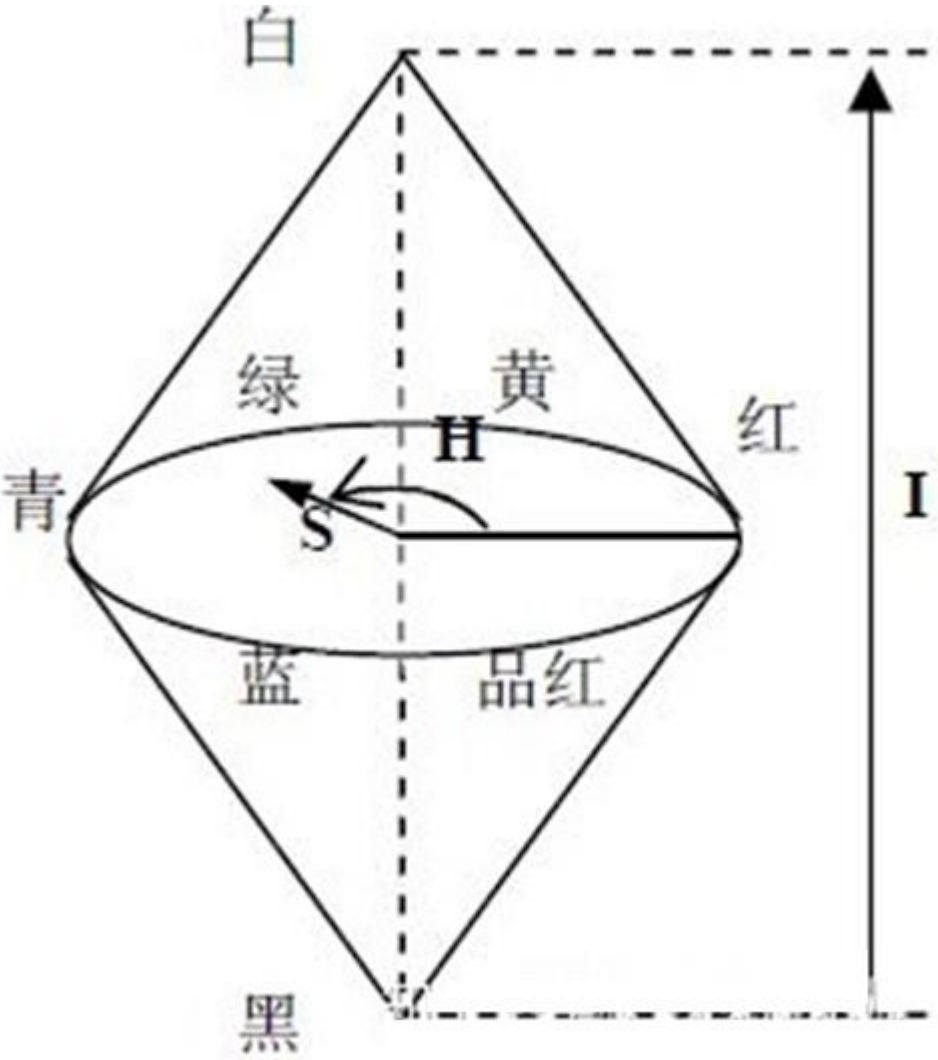
- 颜色模式（色彩模式）：用数字表示颜色的一种算法，是用来显示和打印图像的颜色模型。

- 常用的颜色模式有：HSB、RGB、CMYK、LAB、灰度、位图、双色调、索引颜色和多通道等，每个颜色模式都有其特殊的用途，最常用的是RGB模式、CMYK模式以及HIS（或HSB）模式。

3、颜色

- 色相(Hue)：表示颜色的相位角，是彩色最重要的属性，决定颜色的本质。红、绿、蓝分别相隔120度；互补色分别相差180度，即颜色的类别。简单来说表示是蓝色还是粉色或者绿色。
- 饱和度(Saturation)：表示颜色的深浅程度，饱和度越高，颜色越深。与白色的比例有关，白色比例越多，饱和度越低。简单来说就是表示 深红、浅红、粉红 这种不同的深浅程度。
- 亮度(Intensity)：有的时候称为明度（B），表示色彩的明亮程度，人眼对亮度很敏感。简单来说就是颜色比较亮或者比较暗。

3、颜色



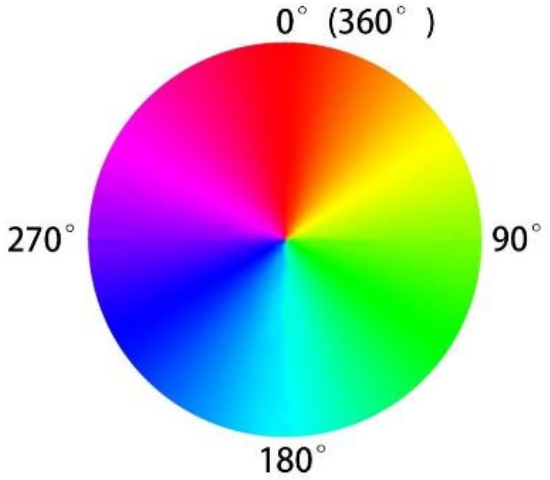
3、颜色

- 色相（H）：色彩可以呈现出来的质地面貌。

（1）通俗的讲就是色彩的相貌或者图像的颜色，比如红、蓝、绿、黄……

（2）色相不是通过百分比，而是以0—360度的角度来表示，它类似一个颜色环，颜色沿着环进行规律性的变化，因此也有12色相环、24色相环等。

（3）黑白灰没有色相。



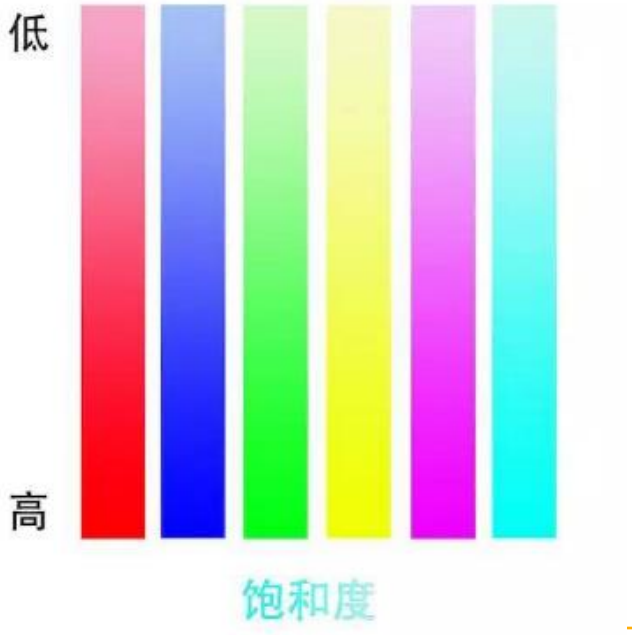
3、颜色

- 饱和度（S）：表示图像颜色的浓度、鲜艳程度。

（1）通俗的讲就是颜色的深浅，如红色可以分为深红、洋红、浅红等。

（2）饱和度高色彩较艳丽，饱和度低色彩就接近灰色。

（3）白、黑和其他灰色色彩都没有饱和度。

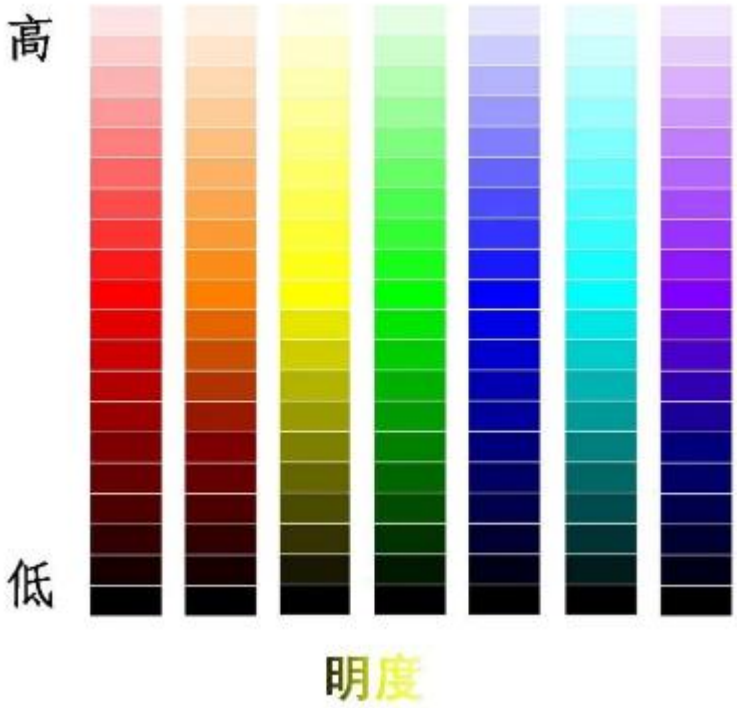


3、颜色

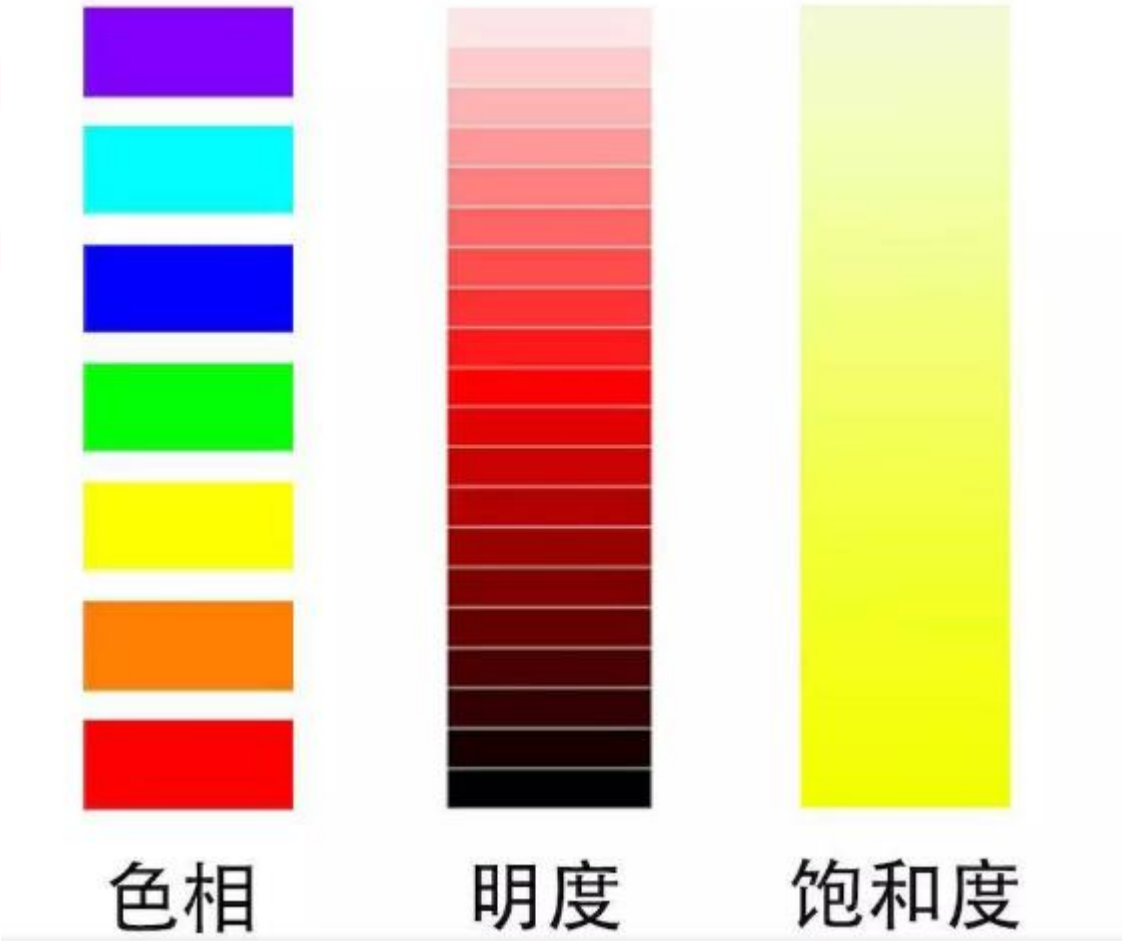
- 亮度（I）：也称为明度（B），指各种颜色的明暗度。

（1）通俗的讲就是表示颜色的强度。

（2）明度高色彩明亮，明度低色彩暗淡，明度最高得到纯白，最低得到纯黑。



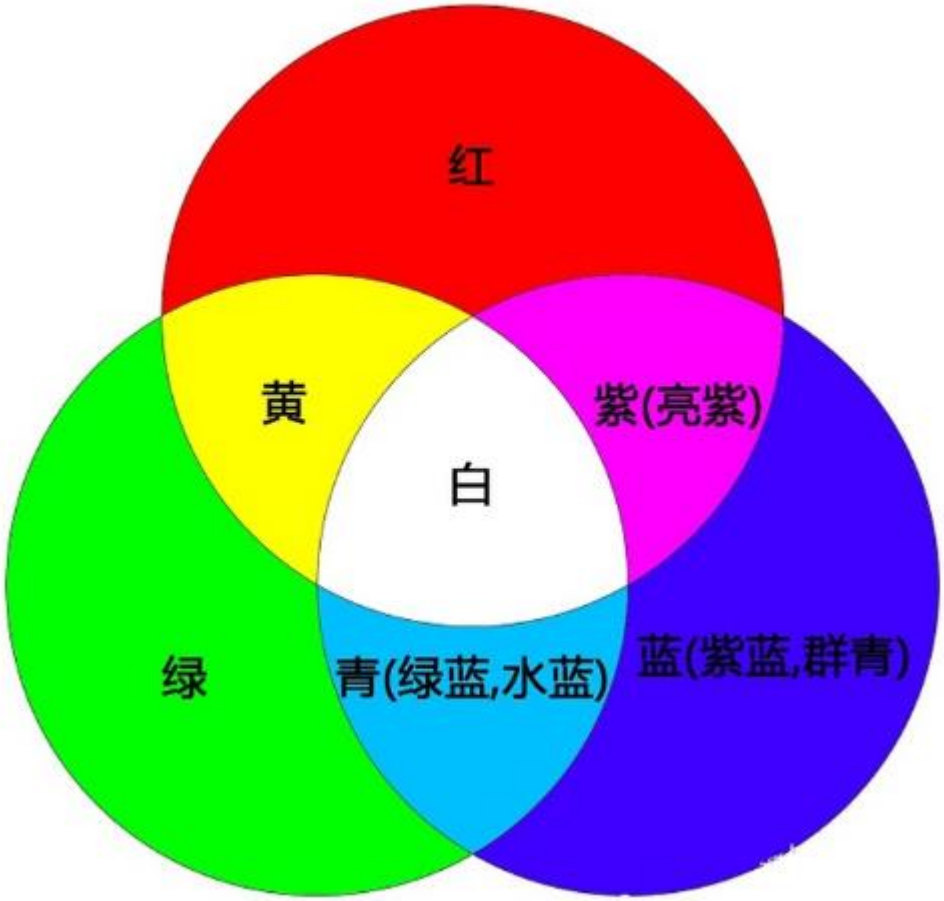
3、颜色



3、颜色

- RGB模式是由红、绿、蓝三种颜色的光线构成的，主要应用于显示器屏幕的显示，因此也被称为色光模式。
- 每一种颜色的光线从0到255被分成256阶，0表示这种光线没有，255表示这种光线饱和。三种光线两两叠加，又形成了青、品、黄三色。黑色表示三种光线都没有，光线越强，颜色越亮，三种光线都饱和即为白色。
- 图像上每个像素使用3个字节来表示颜色值，分别表示 Red、Green、Blue三原色的各个分量的值，由这三原色来构成所有的真彩色效果。对于R、G、B这三个分量我们又通常称之为三个独立的色彩通道。由于每个颜色通道值范围是在[0，255]之间，故RGB通道的3个字节总计可以表示256*256*256=16777216种颜色。
- RGB模式又被称为加色法。

3、颜色



3、颜色

- CMYK模式是由青、品、黄、黑四种颜色的油墨构成的，主要应用于印刷品，因此也被称为色料模式。
- 每一种油墨的使用量从0%到100%，由CMY三种油墨两两混合，又形成了红、绿、蓝三色。由于CMY三种油墨在印刷中无法形成纯正的黑色，因此需要单独的黑色油墨K。油墨量越大，颜色越重、越暗；反之，油墨量越少，颜色越亮，没有油墨时即为白纸。
- CMYK模式又被称为减色法。

4、图像处理技术分类

- 图像处理要求改善图像的视觉效果，针对给定图像的应用场合，有目的地强调图像的整体或局部特性，扩大图像中不同物体特征之间的差别，满足某些特殊分析的需要。其方法是通过一定手段对原图像附加一些信息或变换数据，有选择地突出图像中感兴趣的特征或者抑制（掩盖）图像中某些不需要的特征，使图像与视觉响应特性相匹配。

- 图象处理按所用方法可分成空间域（图象域）法和频率域（变换域）法。前者直接在图象所在空间进行处理，具有代表性的算法包括局部求平均值法和中值滤波（取局部邻域中的中间像素值）法等，可用于去除或减弱噪声；后者把图像看成一种二维信号，对其进行基于二维傅里叶变换的算法处理，如采用低通滤波（即只让低频信号通过）法去掉图中的噪声，或者采用高通滤波法增强图象边缘，使模糊的图片变得清晰。

二、空间域图像增强

1、图像的读取和显示

读取D盘“1”文件夹下的图片：

```
I=imread('D:/示范图片/autumn.tif');
imshow(I) ;
```

2、图像增强——对比度增强

- 由于受到环境，光线等的影响，图像的清晰度和对比度可能比较低，不能够突出图像中的重点，因此图像处理中就给出了图像增强的操作。图像增强就是通过一定手段来增强图像的对比度，使得其中的人物或者事物更加明显，有利于后边的识别等处理。

- 图像增强的基本处理手段是通过基本灰度变化，来实现某种形式的对比度增强。对比度指的是一幅图像中明暗区域最亮的白和最暗的黑之间不同亮度层级的测量，差异范围越大代表对比越大，差异范围越小代表对比越小。

- 对比度对视觉效果的影响非常关键，一般来说对比度越大，图像越清晰醒目，色彩也越鲜明艳丽；而对比度小，则会让整个画面都灰蒙蒙的。

2、图像增强——对比度增强

- 对比度增强常用到的三个matlab函数：imadjust，histeq和adapthisteq。
- imadjust 通过将输入强度图像的值映射到新值来增加图像的对比度，默认情况下，在输入数据的低强度和高强度下，1%的数据处于饱和状态。
- histeq执行直方图均衡化。它通过转换强度图像中的值来增强图像的对比度，以使输出图像的直方图近似匹配指定的直方图（默认情况下为均匀分布）。
- adapthisteq执行对比度限制的自适应直方图均衡。与histeq不同，它在较小的数据区域（图块）而不是整个图像上运行。增强每个图块的对比度，以使每个输出区域的直方图近似匹配指定的直方图（默认情况下为均匀分布）。

2、图像增强——对比度增强

- 图象求反：

灰度级（或RGB值）范围为[0，L－1]的图象，反转后的灰度级（或RGB值）为：t ＝ L－1－s

- 增强对比度

该变化可以将区间[s1，s2]的灰度级（或RGB值）线性映射到区间[t1，t2]上，映射后的灰度级（或RGB值）为：

t＝t1＋（s－s1）＊（t2－t1）／（s2－s1）

2、图像增强——对比度增强

- 对数变换

此变换可以实现图象灰度的压缩或扩散（反函数），绝大多数傅立叶频谱在显示的时候都用这种方法调整过：

t ＝ C*log(1+|s|)，其中C为尺度比例常数。

- 幂次变换

$t = C*r^{\gamma}$ ，其中C为尺度比例常数；γ为正的幂次常数。

随着γ的变化，可以得到一族变换曲线，且γ>1和γ<1的效果正好相反。

例：图像反转

```
I=imread('autumn.tif');
J=double(I);
J=-J+(256-1);                    %图像反转线性变换
H=uint8(J);
subplot(1,2,1),imshow(I);
subplot(1,2,2),imshow(H);
```

例：图像反转

注意：

（1）为了节省存储空间，matlab为图像提供了特殊的数据类型uint8(8位无符号整数），以此方式存储的图像称作8位图像。imread把灰度图像存入一个8位矩阵，若是RGB图像时，就存入8位RGB矩阵中。如果matlab读入图像的数据是uint8，而matlab中数值一般采用double型（64位）存储和运算，就要先将图像转为double格式的才能运算。

```
I2=im2double(I1)  %把图像I1转换成double精度类型 （假设图形矩阵范围0~255）
I64=double(I8)/255;   %uint转换成double
```

如果不转换，计算会产生溢出。

例：图像反转

注意：

（2）经过计算后，I2已经是double型。如果现在想imshow显示图像结果，就需要再转换成uint8格式。如果矩阵符合数据图像标准（0~1之间），则：

```
I3=im2uint8(I2)  %把矩阵I2转换成uint8类型
```

如果超出0~1范围，就要用uint8()处理：

```
I8=uint8(round(I64*255));  %double转换成uint8
```

或者用mat2gray()处理：

```
I3=mat2gray(I2)  &把矩阵转化为灰度图像格式double
```

例：图像对比度增强

```
I=imread('cameraman.tif');
imshow(I);
figure,imhist(I);
J=imadjust(I,[0.3 0.7], [0 1]);
figure,imshow(J);
figure,imhist(J);
```

例：图像对比度增强

注意：

（1）matlab的imread命令很强大，可以读取各种类型的图像。但是imread并不是一个实际功能函数。不同的图像格式有不同的编码方式，因此有不同的读取方式，为每种不同格式的图像编写各自的读取函数才是最适当的，实际中也是这么做的。Matlab同样如此，imread只是一个入口函数。它仅仅是做了一些文件名的处理，从指定的文件名中，找到绝对路径，找到图像后缀名，然后调用合适的读取函数。

（2）imhist函数用于获取图像数据直方图。在图像增强技术中，图像灰度级直方图有着重要的意义，是直方图修改技术、直方图均衡化等一些图像处理技术的基础。

例：图像对比度增强

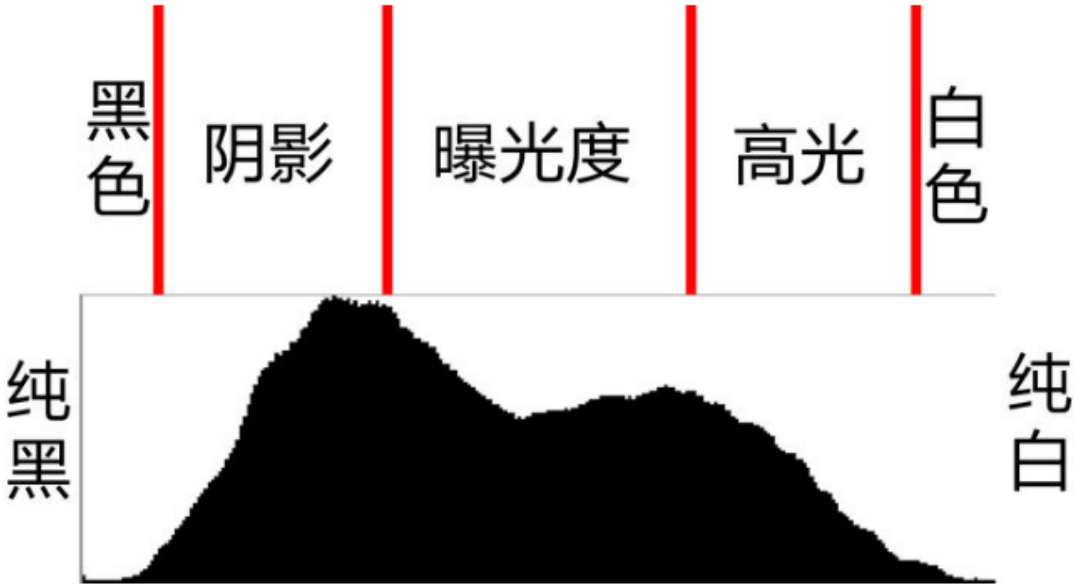
注意：

（3）J = imadjust(I,[low_in; high_in],[low_out; high_out])

将图像I中的亮度值映射到J中的新值，即将low_in至high_in之间的值映射到low_out至high_out之间的值。low_in 以下与high_in 以上的值被剪切掉了，也就是说，low_in 以下的值映射到 low_out，high_in 以上的值映射到high_out。它们都可以使用空的矩阵[]，默认值是[0 1]。

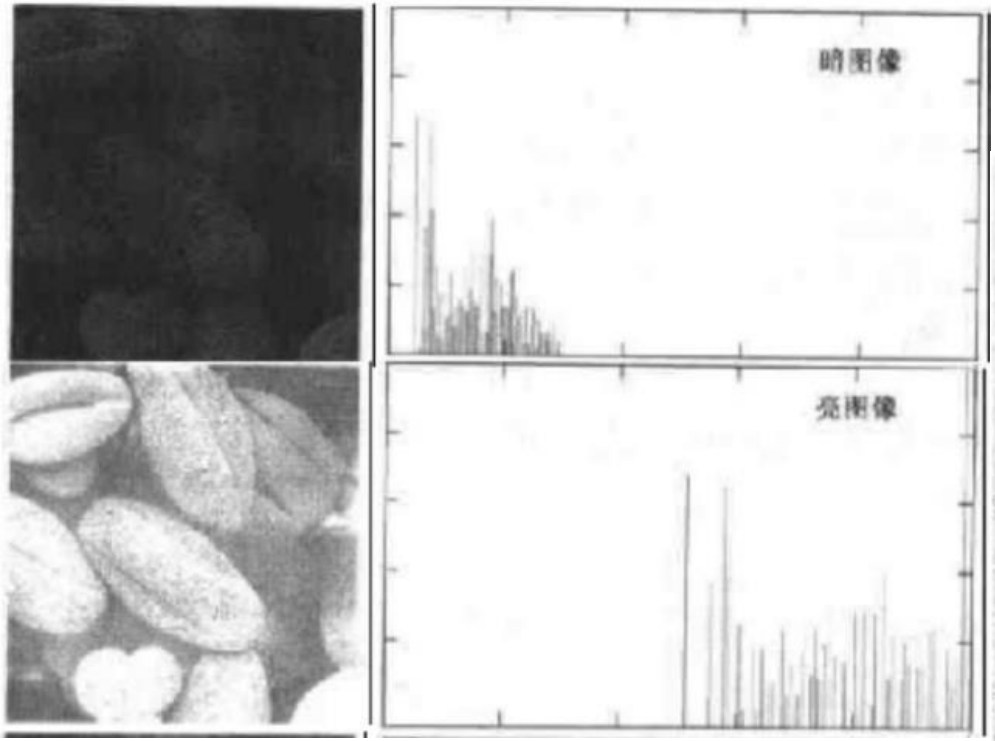
3、直方图处理

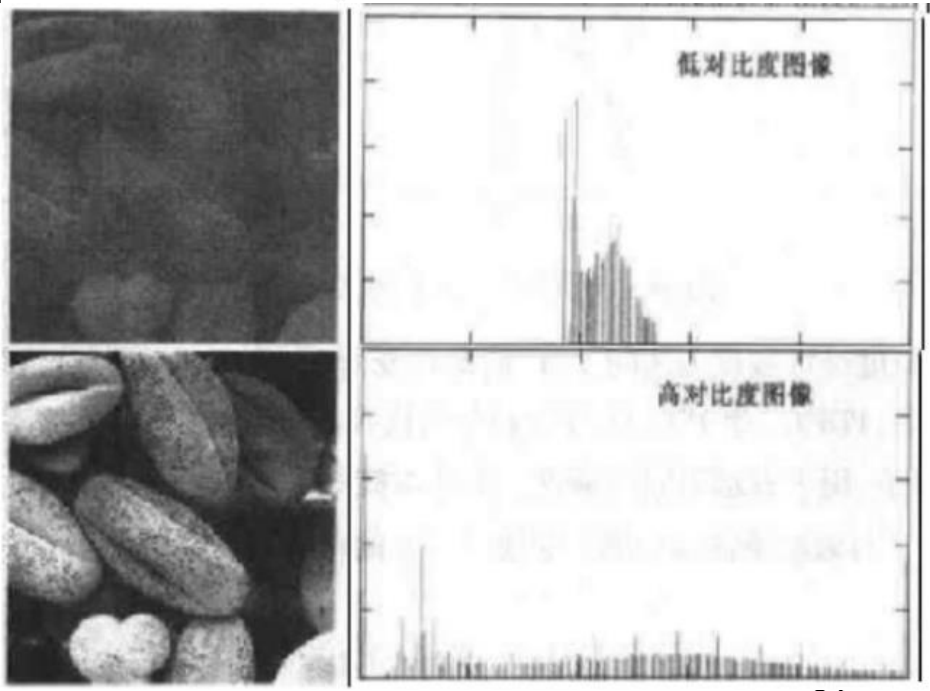
- 以灰度图像为例，我们把图像亮度（灰度）分为0到255共256个数值，数值越大，代表的亮度越高。其中0代表纯黑色的最暗区域，255表示最亮的纯白色，而中间的数字就是不同亮度的灰色。
- 用横轴代表0-255的亮度数值，竖轴代表图像中对应亮度的像素数量，这个函数图像就被称为直方图。直方图中柱子的高度，代表了画面中有多少像素是那个亮度。比如下面的直方图，波峰在中间偏左的位置（阴影区域），说明画面中有很多深灰或者深色部分。



3、直方图处理

- 我们找了四幅图，分别反映了一个花粉图像的四个基本灰度级特征（依次为暗，亮，低对比度，高对比度），图的右侧对应了每幅图像各自的直方图





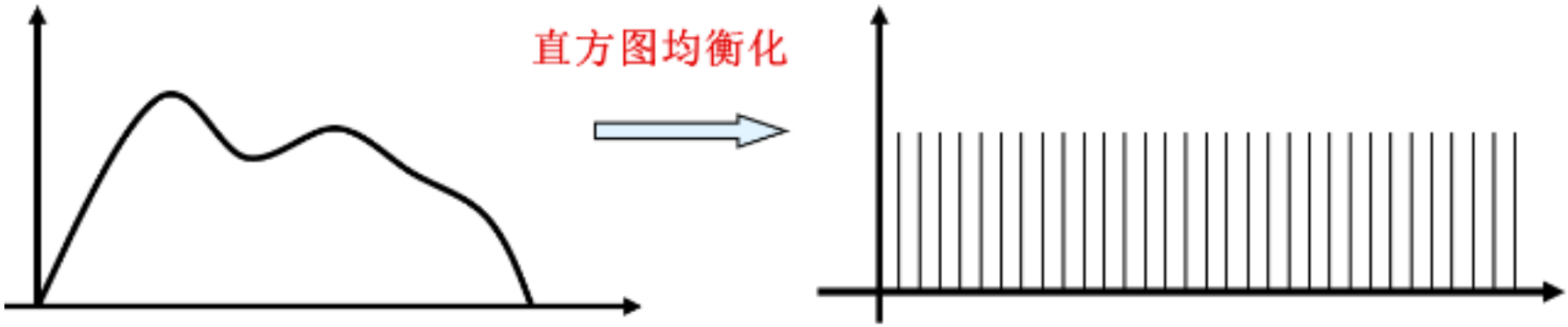
3、直方图处理

- 通过观察上述的四幅图，我们发现：在较暗的图中，直方图的组成成分集中在灰度级低的一侧;在比较亮的图中，直方图的组成成分集中在了灰度级高的一侧;在低对比度的图中，直方图窄而且集中在中间的区域;在高对比度的图中，直方图覆盖了灰度级很宽的范围，而且像素的分布没有太不均匀那么.通过上面的结论可以发现：如果要更加清晰的分辨一幅灰度图，是不是得尽可能保证该图像的直方图尽可能覆盖全部可能的灰度级并且分布均匀呢？这样的图像是不是就会有了高对比度和多变的灰度色调呢？这就是直方图的均衡化的初衷。
- 直方图均衡化就是去开发一种变换函数，使得其对原图进行变换后，得到的新的图像能够满足上面所说的直方图的分布要求。

3、直方图处理

- 直方图均衡化是将原图像通过某种变换，得到一幅灰度直方图为均匀分布的新图像的方法。

- 直方图均衡化方法的基本思想是对在图像中像素个数多的灰度级进行展宽，而对像素个数少的灰度级进行缩减。从而达到清晰图像的目的。



3、直方图处理

- 直方图均衡化

这个方法的基本思想是把原始图象的直方图变换为均匀分布的形式，增加象素灰度值的动态范围，从而达到增强图象整体对比度的效果。

设原始图像在（x，y）处的灰度为s，改变后为t，图像的映射函数定义为：t = EH（s），函数EH（s）必须满足两个条件（其中L为图像的灰度级数）：

EH（s）在0≤s≤L－1范围内是一个单值单增函数，以保证增强处理没有打乱原始图像的灰度排列次序，原图各灰度级在变换后仍保持从黑到白(或从白到黑)的排列。

对于0≤s≤L－1，有0≤t≤L－1，以保证变换前后灰度值动态范围的一致性。

3、直方图处理

- 直方图规定化（直方图匹配）

直方图规定化就是有选择的增强某个灰度范围内的对比度，或使图像灰度值的分布满足特定的条件。该方法主要有三个步骤：

（1）对原始图象的直方图进行灰度均衡化。

（2）规定需要的直方图。

（3）将第一个步骤得到的变换反转过来，将原始直方图对应映射到规定的直方图中。

例：直方图均匀化

```
I=imread('pout.tif');% 读取MATLAB自带的potu.tif图像
imshow(I);
figure,imhist(I);
[J,T]=histeq(I,64);% 图像灰度扩展到0~255，但是只有64个灰度级
figure,imshow(J);
figure,imhist(J); % 获取并显示出图像数据的直方图
figure,plot((0:255)/255,T); % 转移函数的变换曲线
J=histeq(I,32);
figure,imshow(J); % 图像灰度扩展到0~255，但是只有32个灰度级
figure,imhist(J);
```

例：直方图均匀化

注意：

（1）histeq（I，hgram）是用来做直方图均衡的，hgram是均衡化后的灰度级个数。比如histeq(I，16)，就是希望均衡化后的直方图只有16个灰度级；histeq(I，64)，就是希望均衡化后的直方图只有64个灰度级。

假设原始图像为256级，共有X个像素，现希望均衡化为N个灰度级，过程如下：

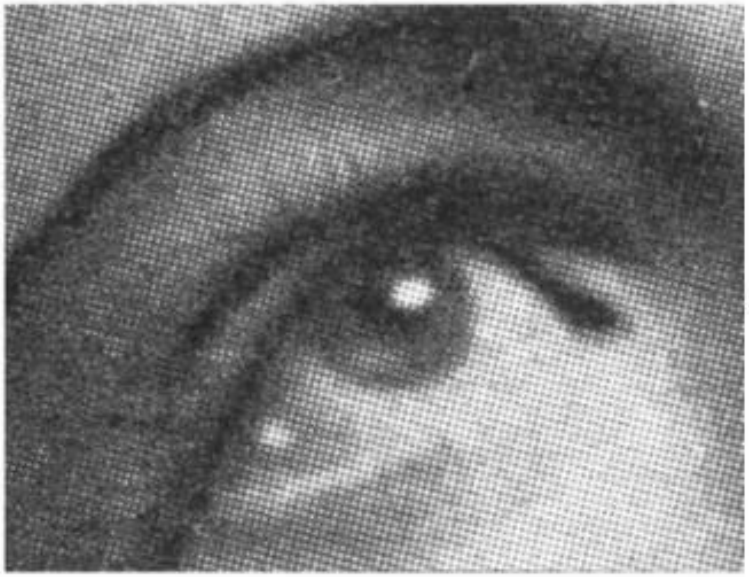
步骤1：计算原始图像I的累积函数A(g)，其中g=[0,255]表示256个灰度级。A(0)就是灰度级为0的元素个数；A(1)就是灰度级为0和1的元素个数和；...；A(255)就是原始图像的总的像素个数。

步骤2：计算原始直方图和均衡化后直方图的映射关系g->G，即将原始图像中的灰度值g变成新的灰度值G。这里，G=N*A(g)/X

步骤3：遍历原始图像I，依据步骤2中的映射关系，把所有像素点的灰度值变成新的灰度值。

4、平滑（低通）空间滤波器

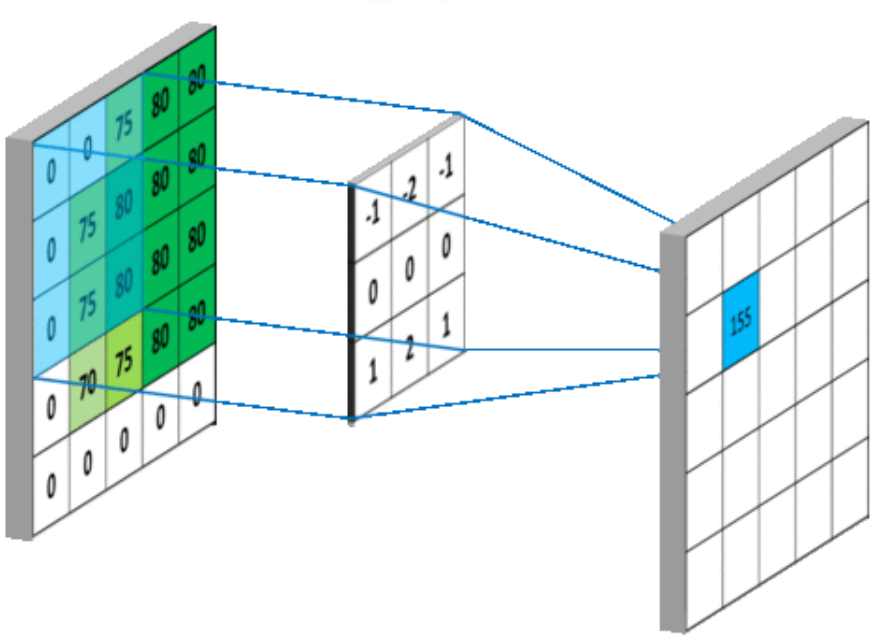
- 算子：对图像的处理函数（如平滑滤波，或者边缘提取等），也往往可以用一个小矩阵来表示，这样的一个矩阵就是一个算子。

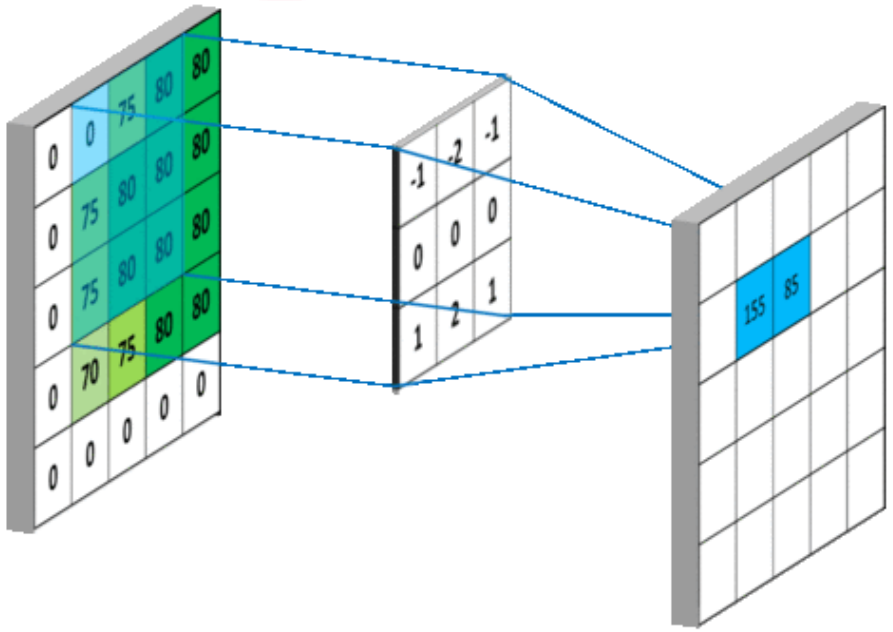


$$\Longrightarrow \begin{bmatrix} a_{0,0} & a_{0,1} & a_{0,2} & \cdots & a_{0,n} \\ a_{1,0} & a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,0} & a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{m,0} & a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{bmatrix}$$

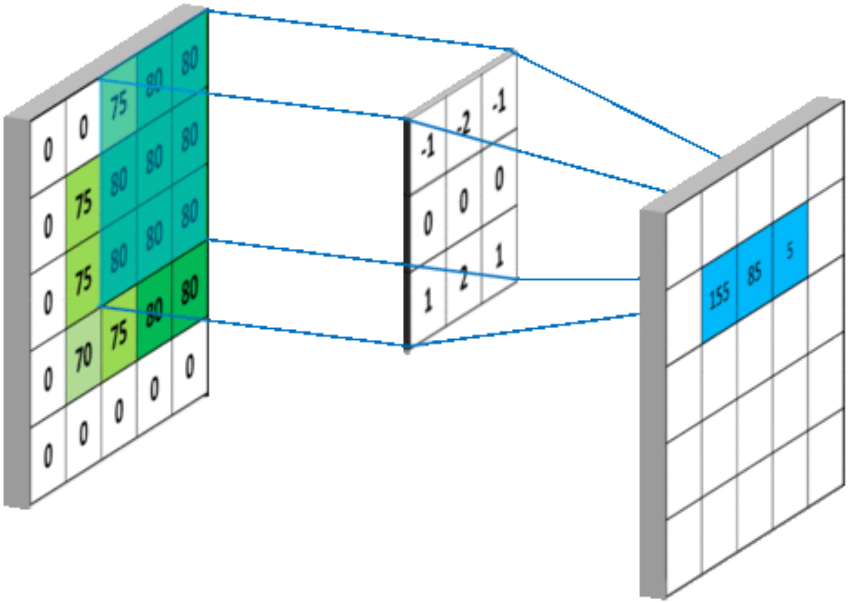
$$g = \begin{bmatrix} b_{-1,-1} & b_{-1,0} & b_{-1,1} \\ b_{0,-1} & b_{0,0} & b_{0,1} \\ b_{1,-1} & b_{1,0} & b_{1,1} \end{bmatrix}$$

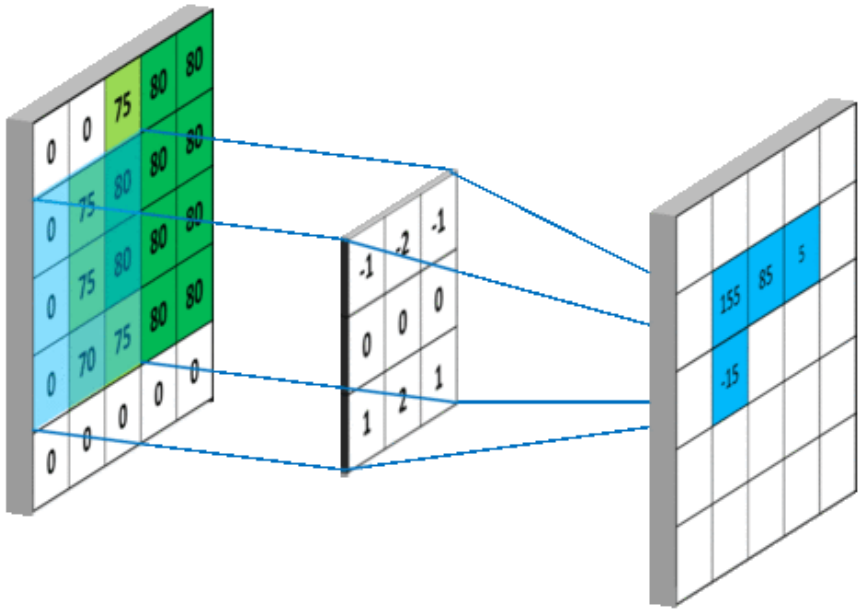
4、平滑（低通）空间滤波器


0 0 75 80 80
0 75 80 80 80
0 75 80 80 80
0 70 75 80 80
0 0 0 0 0
-1 -2 -1
0 0 0
1 2 1
155

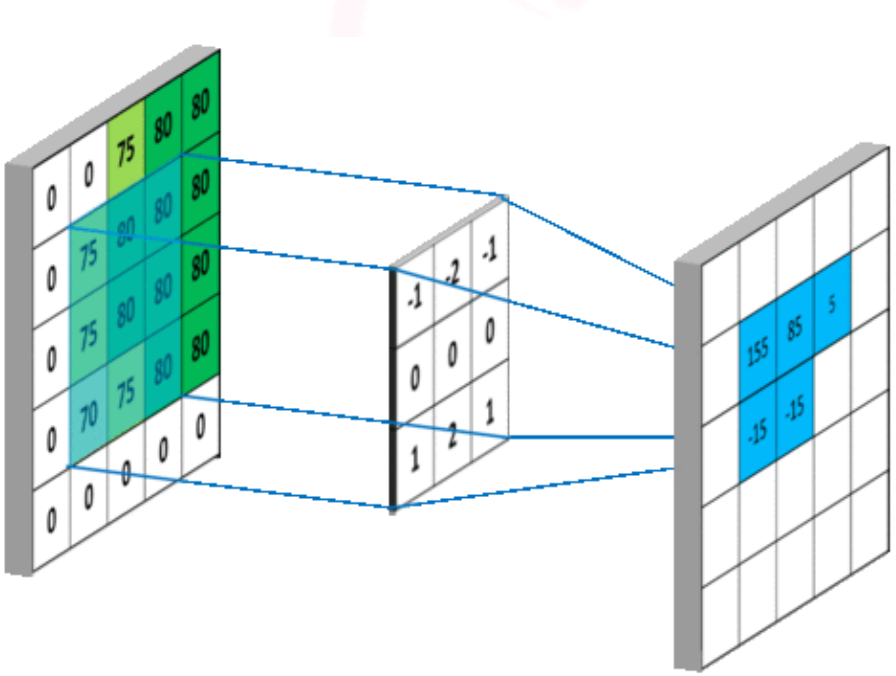

0 0 75 80 80
0 75 80 80 80
0 75 80 80 80
0 70 75 80 80
0 0 0 0 0
-1 -2 -1
0 0 0
1 2 1
155 85

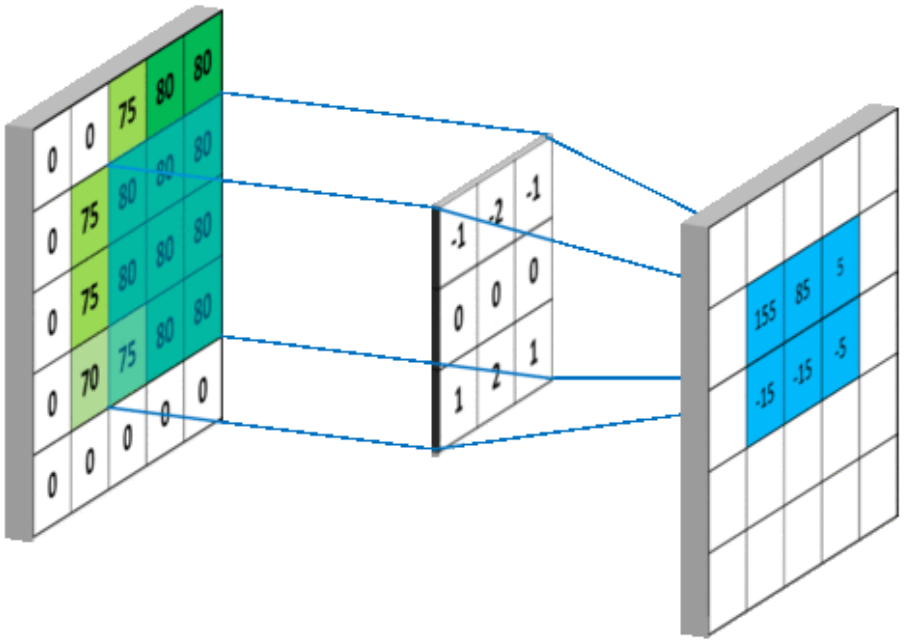
4、平滑（低通）空间滤波器


0 0 75 80 80
0 75 80 80 80
0 75 80 80 80
0 70 75 80 80
0 0 0 0 0
-1 -2 -1
0 0 0
1 2 1
155 85 5

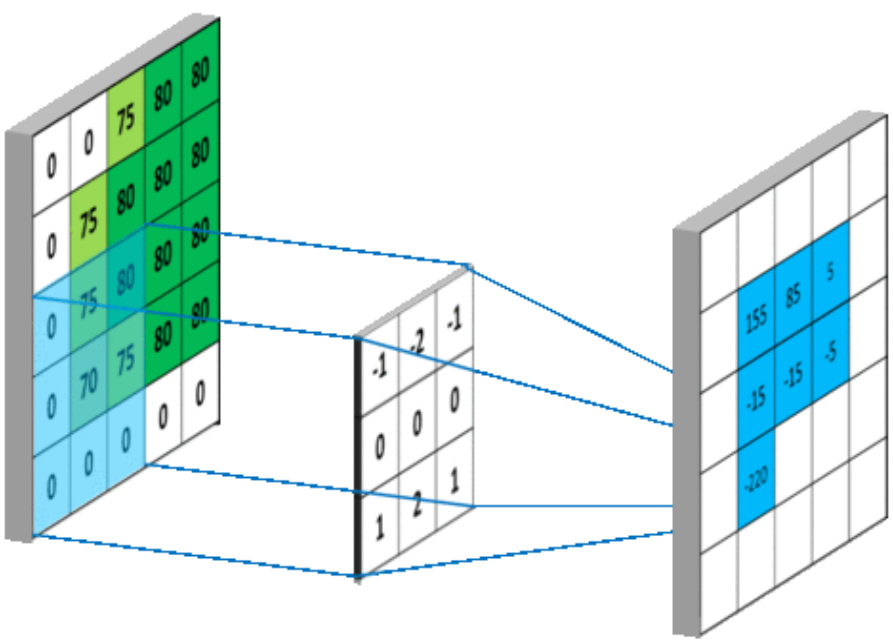

0 0 75 80 80
0 75 80 80 80
0 75 80 80 80
0 70 75 80 80
0 0 0 0 0
-1 -2 -1
0 0 0
1 2 1
155 85 5
-15

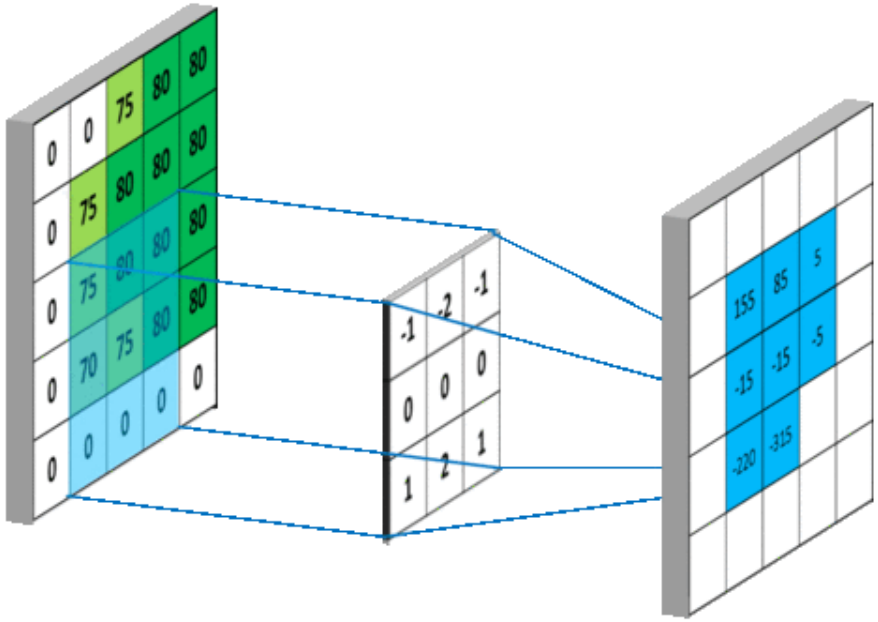
4、平滑（低通）空间滤波器


0 0 75 80 80
0 75 80 80 80
0 75 80 80 80
0 70 75 80 80
0 0 0 0 0
-1 -2 -1
0 0 0
1 2 1
155 85 5
-15 -15

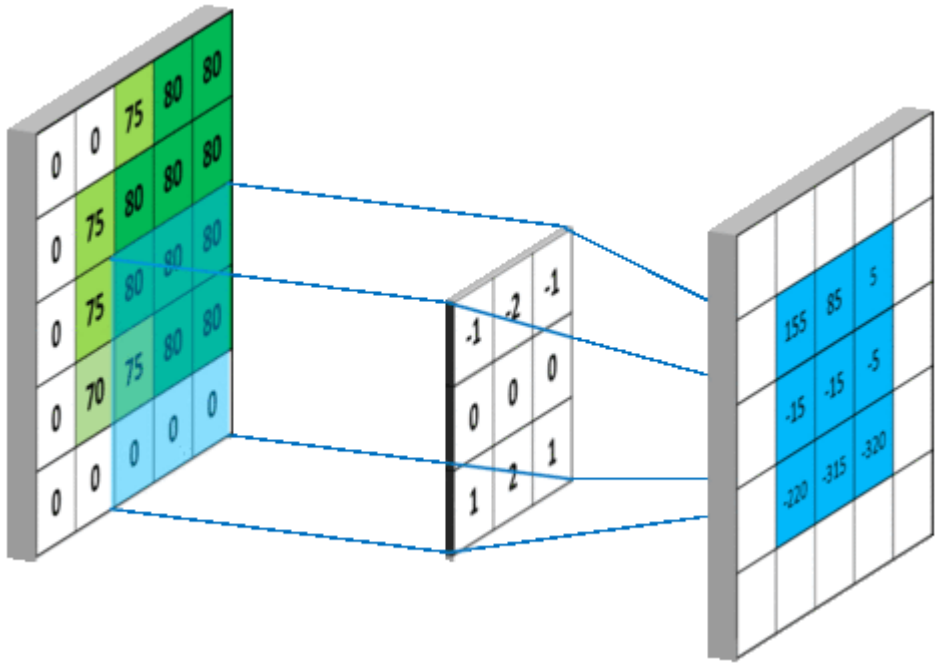

0 0 75 80 80
0 75 80 80 80
0 75 80 80 80
0 70 75 80 80
0 0 0 0 0
-1 -2 -1
0 0 0
1 2 1
155 85 5
-15 -15 -5

4、平滑（低通）空间滤波器


0 0 75 80 80
0 75 80 80 80
0 75 80 80 80
0 70 75 80 80
0 0 0 0 0
-1 -2 -1
0 0 0
1 2 1
155 85 5
-15 -15 -5
-220


0 0 75 80 80
0 75 80 80 80
0 75 80 80 80
0 70 75 80 80
0 0 0 0 0
-1 -2 -1
0 0 0
1 2 1
155 85 5
-15 -15 -5
-220 -315

4、平滑（低通）空间滤波器


0 0 75 80 80
0 75 80 80 80
0 75 80 80 80
0 70 75 80 80
0 0 0 0 0
-1 -2 -1
0 0 0
1 2 1
155 85 5
-15 -15 -5
-220 -315 -320

4、平滑（低通）空间滤波器

- 均值滤波器

采用邻域平均法的均值滤波器非常适用于去除通过扫描得到的图象中的颗粒噪声。领域平均法有力地抑制了噪声，同时也由于平均而引起了模糊现象，模糊程度与邻域半径成正比。

几何均值滤波器所达到的平滑度可以与算术均值滤波器相比，但在滤波过程中会丢失更少的图象细节。

4、平滑（低通）空间滤波器

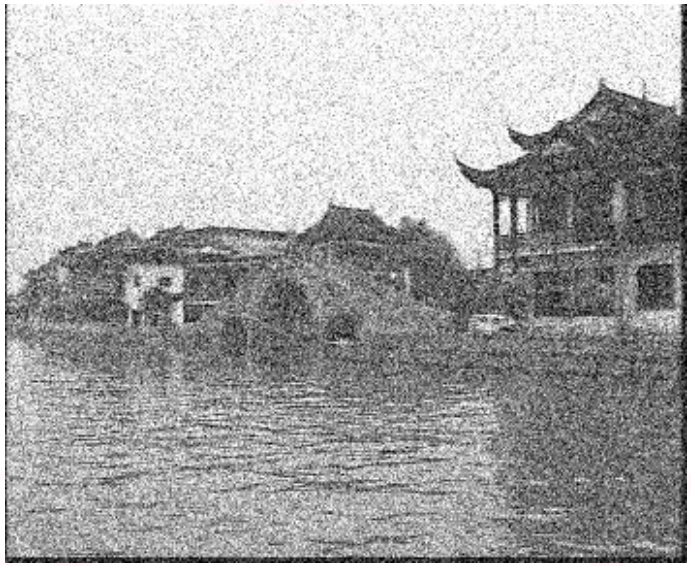
均值滤波器模板如下：

$$T^3=\frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}\qquad T_c^{\,3}=\frac{1}{5}\begin{bmatrix}0&1&0\\1&1&1\\0&1&0\end{bmatrix}$$

$$T^5=\frac{1}{25}\begin{bmatrix}1&1&1&1&1\\1&1&1&1&1\\1&1&1&1&1\\1&1&1&1&1\\1&1&1&1&1\end{bmatrix}\qquad T_c^{\,5}=\frac{1}{21}\begin{bmatrix}0&1&1&1&0\\1&1&1&1&1\\1&1&1&1&1\\1&1&1&1&1\\0&1&1&1&0\end{bmatrix}$$

4、平滑（低通）空间滤波器

无噪声图像



高斯噪声图像

4、平滑（低通）空间滤波器

T3邻域平均

T5邻域平均

4、平滑（低通）空间滤波器

- 中值滤波器

中值滤波器是一种常用的非线性平滑滤波器，其基本原理是把数字图像或数字序列中一点的值，用该点的一个领域中各点值的中值代换。其主要功能是让周围象素灰度值的差比较大的像素改取与周围的像素值接近的值，从而可以消除孤立的噪声点，所以中值滤波对于滤除图像的椒盐噪声非常有效。

中值滤波器可以做到既去除噪声又能保护图像的边缘，从而获得较满意的复原效果，而且，在实际运算过程中不需要图象的统计特性，这也带来不少方便。但对一些细节多，特别是点、线、尖顶细节较多的图象不宜采用中值滤波的方法。

4、平滑（低通）空间滤波器

椒盐噪声图像

3*3中值滤波

4、平滑（低通）空间滤波器

- 高斯滤波器：采用高斯函数作为加权函数。

原因一：二维高斯函数具有旋转对称性，保证滤波时各方向平滑程度相同；

原因二：二维高斯函数可以做到离中心点越远权值越小，确保边缘细节不被模糊。

高斯函数如下：

$$G(x,y)=e^{-\frac{x^2+y^2}{2\sigma^2}}=e^{-\frac{r^2}{2\sigma^2}}$$

4、平滑（低通）空间滤波器

- 高斯滤波器：设计离散高斯滤波器的σ和n，确定高斯模板权值。如σ^2=2和n=5，则：

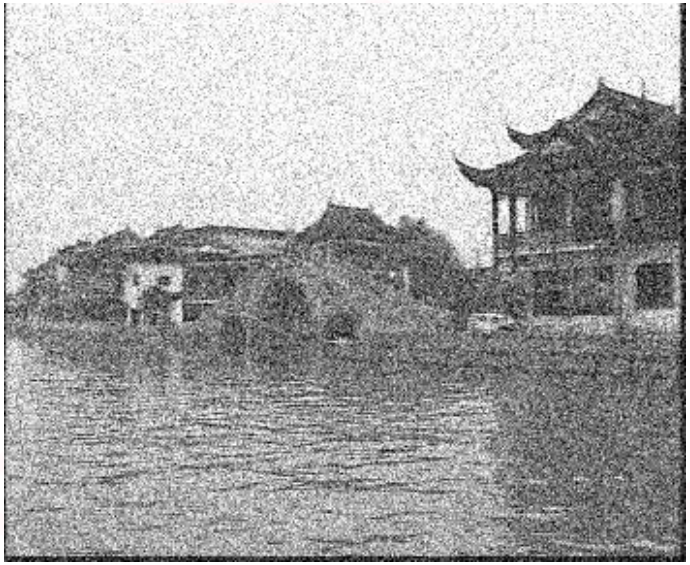
[i,j]	-2	-1	0	1	2
-2	0.135	0.287	0.105	0.287	0.135
-1	0.287	0.606	0.779	0.606	0.287
0	0.105	0.779	1	0.779	0.105
1	0.287	0.606	0.779	0.606	0.287
2	0.135	0.287	0.105	0.287	0.135

4、平滑（低通）空间滤波器

- 高斯滤波器：整数化和归一化后，可得：

[i,j]	-2	-1	0	1	2
-2	1	2	3	2	1
-1	2	4	6	4	2
0	3	6	7	6	3
1	2	4	6	4	2
2	1	2	3	2	1

4、平滑（低通）空间滤波器



噪声图像

高斯滤波器处理效果

例：高斯白噪声和椒盐噪声

```
I=imread('eight.tif');
imshow(I) ;
J1=imnoise(I,'gaussian',0,0.02); % 叠加均值为0，方差为0.02的高斯噪声，可以用localvar代替
figure,imshow  (J1);
J2=imnoise(I,'salt & pepper',0.04); % 叠加密度为0.04的椒盐噪声。
figure,imshow(J2);
```

例：高斯白噪声和椒盐噪声

注意：

（1）imnoise 是表示添加噪声污染一幅图像，叫做噪声污染图像函数。I为是输入图像，函数imnoise在给图像添加噪声之前，会将它转换为范围[0,1]内的double类图像。指定噪声参数时必须考虑到这一点。

g=imnoise(I,‘gaussian’,m,var)，将均值m、方差为var的高斯噪声加到图像I上，默认值是均值m为0，方差var为0.01的噪声。

g=imnoise（ I,'localvar',V)将均值为0，局部方差为V的高斯噪声添加到图像I上，其中V是与f大小相同的一个数组，它包含了每一个点的理想方差值。

g=imnoise(I,‘localvar’,image_intensity,var)将均值为0的高斯噪声添加到图像I中，其中噪声的局部方差var是图像I的亮度值的函数。向量image_intensity必须包含范围在[0,1]内的归一化亮度值。

例：高斯白噪声和椒盐噪声

注意：

g=imnoise(I,'salt&pepper',d)用椒盐噪声污染图像I，其中d是噪声密度（即包括噪声值的图像区域的百分比）。因此，大约有d*numel（I）个像素受到影响。默认的噪声密度为0.05。

g=imnoise(I,'speckle',var)用方程g=I+n*I将乘性噪声添加到图像f上，其中n是均值为0，方差为var的均匀分布的随机噪声，var的默认值是0.04。

g=imnoise(I,'poisson')从数据中生成泊松噪声，而不是将人工的噪声添加到数据中，为了遵守泊松统计，unit8和unit16类图像的亮度必须和光子的数量相符合。当每个像素的光子数量大于65535时，就要使用双精度图像。亮度值在0到1之间变化，并且对应于光子的数量除以10e12。

例：中值滤波

```
I=imread('eight.tif');
imshow(I) ;
J2=imnoise(I,'salt & pepper',0.04); % 叠加密度为0.04的椒盐噪声。
figure,imshow(J2);
I_Filter1=medfilt2(J2,[3 3]);  %窗口大小为3×3
figure,imshow(I_Filter1);
I_Filter2=medfilt2(J2,[5 5]);  %窗口大小为5×5
figure,imshow(I_Filter2);
I_Filter3=medfilt2(J2,[7 7]);  %窗口大小为7×7
figure,imshow(I_Filter3);
```

例：中值滤波

注意：

（1）medfilt2即为中值滤波器，它对于斑点噪声（en:speckle noise）和椒盐噪声（en:salt-and-pepper noise）来说尤其有用。

5、锐化（高通）空间滤波器

- **梯度锐化**

$$f_x = f(i,j) - f(i+1,j) \quad f_y = f(i,j) - f(i,j+1)$$

$$\because f'(x,y) = T * f(x,y) = \sum_{i=0}^{m-1}\sum_{j=0}^{m-1} T(i,j)\, f(x-i,y-j)$$

$\therefore f_x$相当于$f(x,y)$对卷积模板$\begin{bmatrix}1 & -1\end{bmatrix}$做相关（卷积）运算；

f_y相当于$f(x,y)$对卷积模板$\begin{bmatrix}1\\-1\end{bmatrix}$做相关（卷积）运算；

但是此时两点分别位于$\left(i+\frac{1}{2}, j\right)$和$\left(i, j+\frac{1}{2}\right)$处，

因此常分别采用2×2模板：

$\begin{bmatrix}1 & -1\\1 & -1\end{bmatrix}$和$\begin{bmatrix}1 & 1\\-1 & -1\end{bmatrix}$，此时梯度点位于$\left(i+\frac{1}{2}, j+\frac{1}{2}\right)$处。

5、锐化（高通）空间滤波器

- 边缘检测算子

（1）Roberts算子：

$$\varphi=\sqrt{f_x^{\,2}+f_y^{\,2}}\approx|f_x|+|f_y|$$

$$=|f(i,j)-f(i+1,j+1)|+|f(i+1,j)-f(i,j+1)|$$

其卷积模板分别是：

$$H_1=\begin{bmatrix}0 & 1\\ -1 & 0\end{bmatrix}\quad H_2=\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}$$

Roberts算子特点是边缘定位准，对噪声敏感。

5、锐化（高通）空间滤波器

- 边缘检测算子

（2）Prewitt算子：采用3x3模板，平均、微分操作对噪声有抑制作用。

(i,j)象素点梯度近似计算为

$$\begin{matrix} a_0 & a_1 & a_2 \\ a_7 & (i,j) & a_3 \\ a_6 & a_5 & a_4 \end{matrix}$$

$$f_x=(a_2+a_3+a_4)-(a_0+a_7+a_6)$$

$$f_y=(a_6+a_5+a_4)-(a_0+a_1+a_2)$$

$$\therefore H_1=\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad H_2=\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

5、锐化（高通）空间滤波器

- 边缘检测算子

（3）Sobel算子：与Prewitt算子类似，但采用了加权。

$$H_1=\begin{bmatrix}-1 & 0 & 1\\ -2 & 0 & 2\\ -1 & 0 & 1\end{bmatrix}\quad H_2=\begin{bmatrix}-1 & -2 & -1\\ 0 & 0 & 0\\ 1 & 2 & 1\end{bmatrix}$$

Isotropic Sobel算子：

$$H_1=\begin{bmatrix}-1 & 0 & 1\\ -\sqrt{2} & 0 & \sqrt{2}\\ -1 & 0 & 1\end{bmatrix}\quad H_2=\begin{bmatrix}-1 & -\sqrt{2} & -1\\ 0 & 0 & 0\\ 1 & \sqrt{2} & 1\end{bmatrix}$$

Sobel算子在实际中最常用。

5、锐化（高通）空间滤波器

Lenna图

Sobel处理效果

5、锐化（高通）空间滤波器

Prewitt处理效果

Roberts处理效果

5、锐化（高通）空间滤波器

- **拉普拉斯锐化：拉普拉斯算子是不依赖边缘方向的二阶微分算子，是一个标量而不是一个向量，具有旋转不变性即各向同性的性质。**

$$\nabla^2 f(x,y)=\frac{\partial^2 f(x,y)}{\partial x^2}+\frac{\partial^2 f(x,y)}{\partial y^2}$$

$$\partial f(x,y)/\partial x = f(x+1,y)-f(x,y)$$

$$\partial^2 f(x,y)/\partial x^2 = f(x+1,y)-f(x,y)-f(x,y)+f(x-1,y)$$

近似为$\nabla^2 f(x,y)=f(x+1,y)+f(x-1,y)+$

$f(x,y+1)+f(x,y-1)-4f(x,y)$

5、锐化（高通）空间滤波器

- **拉普拉斯锐化的卷积模板为：**

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

有时希望邻域中心点具有更大的权值

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$ 或

$$\begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}$$

拉氏算子在实际应用中对噪声敏感。因此在实际中通常不直接使用。

5、锐化（高通）空间滤波器

- LOG方法：梯度算子和拉普拉斯算子都对噪声敏感，并能够使噪声成分加强。因此，处理含有较大噪声的图像时，往往先对图像进行平滑操作，然后再进行二阶微分，即LOG（又称Marr方法）方法，即：先用高斯函数进行平滑，再用拉普拉斯算子进行锐化。

5、锐化（高通）空间滤波器

LOG方法：Delta=2

LOG方法：Delta=4

5、锐化（高通）空间滤波器

- Canny方法：图象边缘检测必须满足两个条件：一能有效地抑制噪声；二必须尽量精确确定边缘的位置。根据对信噪比与定位乘积进行测度，得到最优化逼近算子，这就是Canny边缘检测算子。Canny方法类似与LoG方法，也属于先平滑后求导数的方法。Canny方法的步骤如下：

  step1：用高斯滤波器平滑图象；

  step2：用一阶偏导的差分来计算梯度的幅值和方向；

  step3：对梯度幅值进行非极大值抑制；

  step4：边缘检测和连接边缘。

5、锐化（高通）空间滤波器

- Canny方法的程序实现过程中，会将上述的4个步骤细分为7步：

生成高斯滤波系数；
用生成的高斯滤波系数对原图像进行平滑；
求滤波后图像的梯度；
进行非极大抑制；
统计图像的直方图，对阈值进行判定；
利用函数寻找边界起点；
根据第6步执行的结果，从一个像素点开始搜索，搜索以该像素点为边界起点的一条边界的一条边界的所有边界点。

5、锐化（高通）空间滤波器

- Canny方法的非极大抑制

Canny边缘检测首先对图像做高斯滤波和生成梯度图，得到的梯度图中，边缘的宽度可能大于一个象素，非极大抑制的过程是沿着边缘的梯度方向进行检测，把不是局部极大值的象素置0，这样得到了只有一个象素宽度的边缘。 其作用是将梯度图中的边缘细化成一个象素宽度。

5、锐化（高通）空间滤波器

Canny方法： τ=2

Canny方法： τ=4

例：Sobel算子和拉普拉斯

```
I=imread('autumn.tif');
subplot(2,2,1),imshow(I);
title('原始图像');
axis([50,250,50,200]);
grid on;                          %显示网格线
axis on;                          %显示坐标系
I1=im2bw(I);
subplot(2,2,2),imshow(I1);
title('二值图像');
axis([50,250,50,200]);
grid on;                          %显示网格线
axis on;                          %显示坐标系
H=fspecial('sobel');        %选择sobel算子
```

例：Sobel算子和拉普拉斯

```
J=filter2(H,I1);                    %卷积运算
subplot(2,2,3),imshow(J);
title('sobel算子锐化图像');
axis([50,250,50,200]);
grid on;                            %显示网格线
axis on;                            %显示坐标系
h=[0 1 0,1 -4 1,0 1 0];             %拉普拉斯算子
J1=conv2(single(I1),single(h),'same');  %卷积运算
subplot(2,2,4),imshow(J1);
title('拉普拉斯算子锐化图像');
axis([50,250,50,200]);
grid on;                            %显示网格线
axis on;                            %显示坐标系
```

例：Sobel算子和拉普拉斯

注意：

（1）函数im2bw可以使用阈值（threshold）变换法把灰度图像（grayscale image）转换成二值图像。所谓二值图像， 一般意义上是指只有纯黑（0）、纯白（255）两种颜色的图像。

（2）J = filter2(H,I1)，使用指定的滤波器H对I1进行滤波，结果保存在J中。

（3）conv2是MATLAB中提供的卷积运算函数，该命令的语法格式为：C = conv2(A,B) ，返回矩阵A和B的二维卷积C。默认值‘full’表示返回全部二维卷积值；‘same’返回与A大小相同卷积值的中间部分；‘valid’指在n维卷积运算中，C的大小为max(size(A)- size(B)+1,0)，若A为ma×na的矩阵，B为mb×nb的矩阵，则C的大小为(ma+mb-1)×(na+nb-1)。

例：Sobel算子和拉普拉斯

注意：

（4） fspecial函数用于建立预定义的滤波算子，其语法格式为：

h = fspecial(type)

h = fspecial(type, para)

其中type指定算子的类型，para指定相应的参数；type包括：

‘average ’ ：均值滤波，参数为hsize，代表模板尺寸，默认值为[3, 3]。

‘disk’：圆形区域均值滤波，参数为radius，代表区域半径，默认值为5。

‘gaussian ’：高斯低通滤波，有两个参数，hsize表示模板尺寸，默认值为[3 3]，sigma为滤波器的标准值，单位为像素，默认值为0.5.

例：Sobel算子和拉普拉斯

注意：

‘laplacian’:拉普拉斯算子，参数alpha用于控制算子形状，取值范围为[0，1]，默认值为0.2。

‘log’：拉普拉斯高斯算子，有两个参数，hsize表示模板尺寸，默认值为[3 3]，sigma为滤波器的标准差，单位为像素，默认值为0.5。

‘motion’：运动模糊算子，有两个参数，表示摄像物体逆时针方向以theta角度运动了len个像素，len的默认值为9，theta的默认值为0。

‘prewitt’：用于边缘增强，大小为[3 3]，无参数。

‘sobel’：用于边缘提取，无参数。

‘unsharp’：为对比度增强滤波器。参数alpha用于控制滤波器的形状，范围为[0，1]，默认值为0.2。

例：Sobel算子和拉普拉斯

注意：

（5）cov2（）这个函数，在matlab2014中做了一些修改，对函数中的参数类型做了限定。以前的matlab使用的代码是：

J1=conv2(I1, h, ‘same’);可以正常运行。但是现在需要把函数中第一个、第二个参数的类型改成单精度值（single）或双精度值（double）。即：

```
J1=conv2(single(I1),single(h),'same');
```

谢谢！