Matlab图像处理编程实践初步

图像增强

肖俊

浙江大学计算机学院

2025

内容提要

- 灰度变换增强
- 图像的空域滤波增强

图像增强

图像增强是一种基本的图像处理技术，可以通过突出图像中的某些信息，同时抑制或去除某些不需要的信息来提高图像的质量。例如，强化图像的高频分量，可使图像中物体的轮廓更为清晰。图像增强不以图像保真为原则，其主要目的有两个。

- 改善图像的视觉效果，提高清晰度。
- 使图像变得更有利于计算机的处理，如通过锐化处理突出图像的边缘轮廓线，以便进行各种特征分析。

根据增强处理的作用域不同，图像增强可以划分为空域增强方法和频域增强方法两大类。空域增强是指直接在像素域进行处理，而频域增强是指在图像的变换域内对图像进行处理，然后反变换后即可得到增强以后的图像。此外，还有一些特殊的图像增强，称为彩色增强，包括伪彩色增强和真彩色增强。针对不同的图像应采用不同的图像增强方法，或者同时采用几种适当的增强算法进行实验，从中选出视觉效果好、计算量适中的算法。因此，图像增强技术大多数属于试探式和面向问题的。

灰度变换增强

灰度变换增强是根据某种目标条件，按一定变换关系逐点改变原图像中每一个像素点的灰度值的方法。假设原图像像素的灰度值为 $D=f(x,y)$，处理后图像像素的灰度值为 $D'=g(x,y)$，则灰度增强可表示为，

$$f(x,y)=T[g(x,y)]$$

$$D'=T(D)$$

通过变换，达到对比度增强的效果。当灰度变换关系 $D'=T(D)$ 确定后，则确定一个具体的灰度增强法。$D'=T(D)$ 通常是一个单值函数。

灰度变换增强——线性变换

1. 线性变换

假定原图像 $f(x,y)$ 的灰度范围为 $[a,b]$，希望变换后图像 $g(x,y)$ 的灰度范围扩展至 $[c,d]$，则线性变换的表达式为：

$$g(x,y)=[(d-c)/(b-a)](f(x,y)-a)+c \qquad (6\text{-}1)$$

此关系可用图 6-5 表示。

如果图像中大部分像素的灰度级分布在区域[a,b]之间，小部分灰度级超出了此区域，为改善增强效果，可以用如下所示的变换关系：

$$g(x,y)=\begin{cases} c, & 0\leqslant f(x,y)<a \\ \dfrac{d-c}{b-a}[f(x,y)-a]+c, & a\leqslant f(x,y)<b \\ d, & b\leqslant f(x,y)\leqslant M \end{cases} \qquad (6\text{-}2)$$

此关系可用图 6-6 表示。

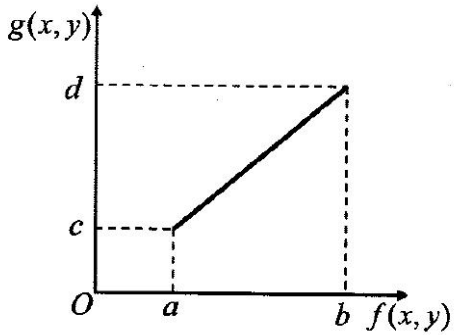


图 6-5　灰度范围线性变换关系

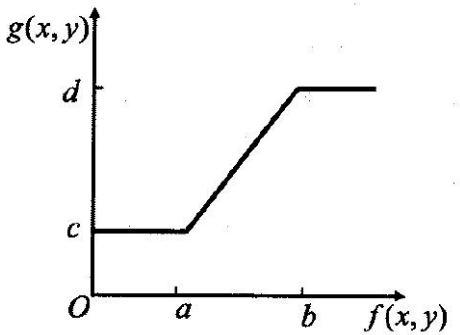


图 6-6　线性变换关系

灰度变换增强——线性变换

2. 分段线性变换

在图像增强中，为了突出感兴趣的目标或灰度区间，相对抑制那些不感兴趣的灰度区间，可以采用分段线性变换。常用的方法是分 3 段作线性变换，如图 6-11 所示。其数学表达式为：

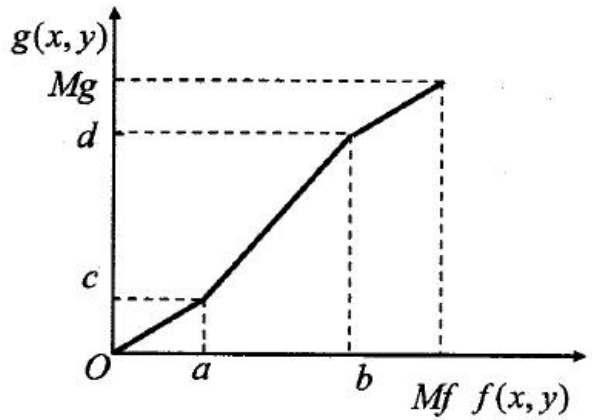


图 6-11　分段线性变换关系

$$g(x,y)=\begin{cases}(c/a)f(x,y), & 0\leqslant f(x,y)<a\\ [(d-c)/(b-a)][f(x,y)-a]+c, & a\leqslant f(x,y)<b\\ [(M_g-d)/(M_f-b)][f(x,y)-b]+d, & b\leqslant f(x,y)\leqslant M_f\end{cases}\qquad(6\text{-}3)$$

图中对灰度区间$[a,b]$进行线性变换，而灰度区间$[0,a]$和$[b,M_f]$受到了压缩。通过调整折线拐点的位置及控制分段直线的斜率，可对任意灰度区间进行扩展或压缩。这种变换适用于在黑色或白色附近有噪声干扰的情况。

灰度变换增强——非线性变换

3. 非线性灰度变换

当用某些非线性函数，例如对数函数作为图像的映射函数时，可实现图像灰度的非线性变换。对数变换的一般形式为：

$$g(x,y)=a+\frac{\ln[f(x,y)+1]}{b\cdot\ln c} \tag{6-4}$$

式中，a、b、c是便于调整曲线的位置和形状而引入的参数。

对数变换使低灰度范围的f得以扩展，而高灰度范围的f得到压缩，以使图像分布均匀，与人的视觉特性相匹配。图像的对数变换关系如图 6-13 所示。

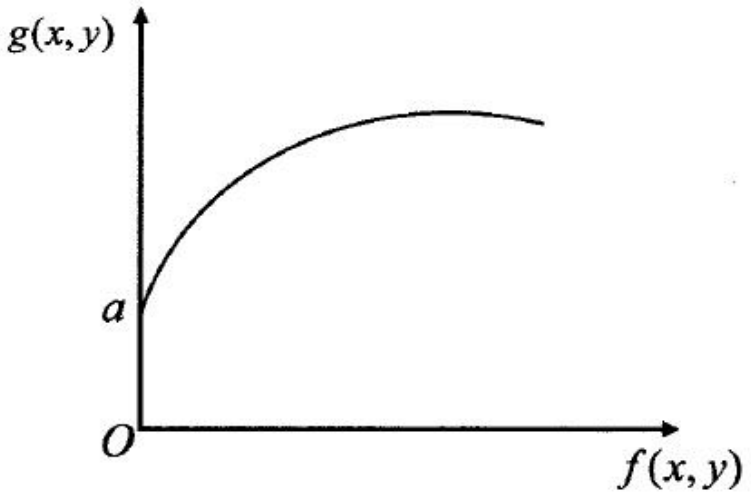
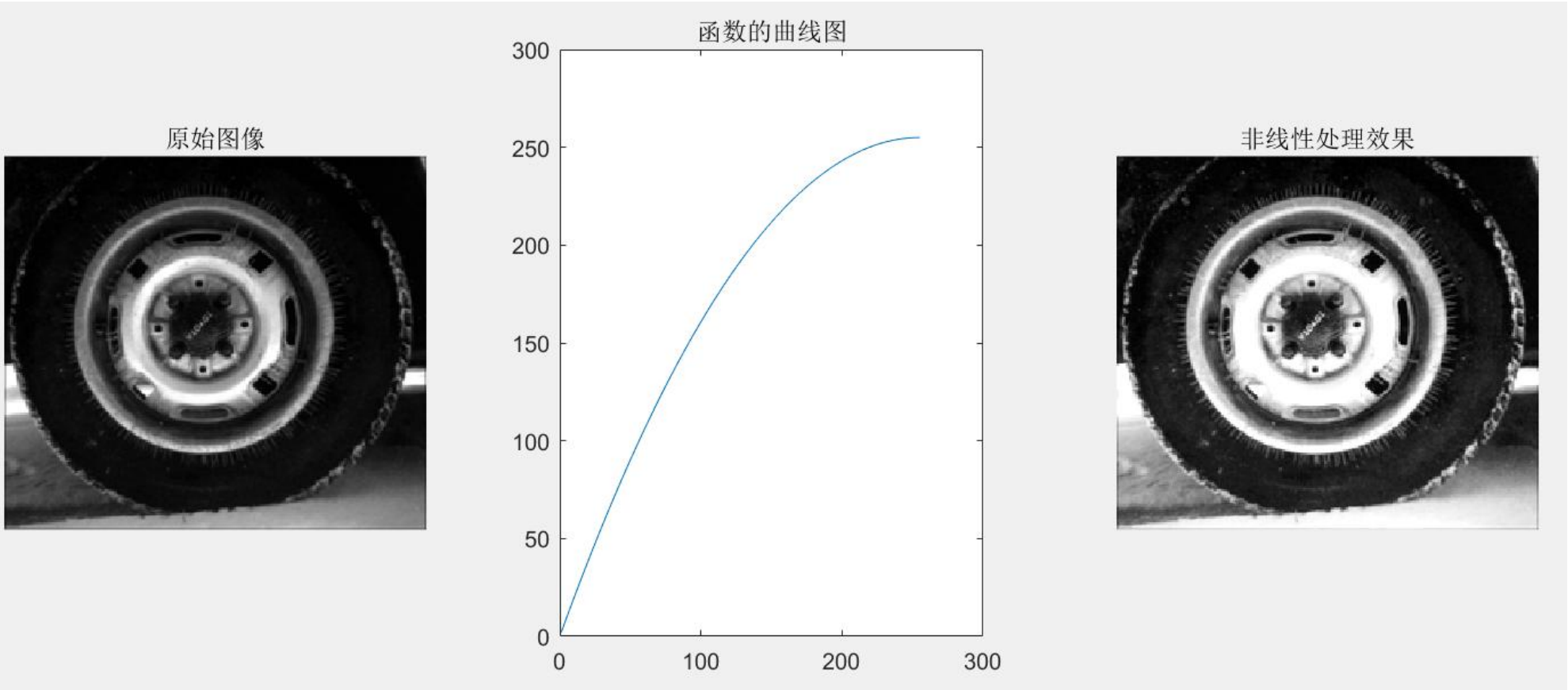


图 6-13　图像的对数变换关系

灰度变换增强——非线性变换

```
a=imread('tire.tif');          %读取原始图像
subp
imsh
title('
%显
x=1:
y=x+
subp
plot(
title('
b1=d
subp
imsh
title('非线性处理效果');
```



灰度变换增强——灰度变换函数

在 MATLAB 图像处理工具箱中提供了有关对比度增强（灰度变换）函数，分别为 imadjust 函数与 brighten 函数。

1. imadjust 函数

imadjust 函数用于调整灰度值或色图。其调用格式为：

（1）J = imadjust(I)：将灰度 I 转换成图像 J。

（2）J = imadjust(I,[low_in; high_in],[low_out; high_out])：将灰度 I 转换成图像 J，值在[low_in; high_in]间的值对应于[low_out; high_out]，其他的被剪切，并且把 low_in、high_in 分别置换成与 low_out 和 high_out 相对应，默认值为[0,1]。

（3）J = imadjust(I,[low_in; high_in],[low_out; high_out],gamma)：参数 gamma 用来描述变换函数的曲线形状。gamma<1 时，增强亮度；gamma>1 时，增强暗度；默认 gamma=1，表示线性变换。

（4）newmap = imadjust(map,[low_in; high_in],[low_out; high_out],gamma)：对索引图像进行变换。如果 low_in、high_in、low_out、high_out 和 gamma 为标量，和上面灰度变换一样，分别对红、绿和蓝三色进行变换。如果 low_in、high_in、low_out 和 high_out 为三元向量或 gamma 为三元向量，映射不同的颜色（三色）分别进行各自指定的变换。newmap 和 map 大小一致。

（5）RGB2 = imadjust(RGB1,...)：分别对每个图像进行对应映射变换。

灰度变换增强——灰度变换函数

```
clear
I=imread('glass.png');                    %读取图像
subplot(2,2,1);
imshow(I);
title('原图像');
subplot(2,2,2);
imhist(I);
title('原图像直方图');
subplot(2,2,3);
J=imadjust(I,[],[0.4 0.6]);               %调整图像的灰度到指定范围
imshow(J);
title('调整灰度后的图像');
subplot(2,2,4);
imhist(J);
title('调整灰度后的直方图');
```

灰度变换增强——灰度变换函数

2. brighten 函数

brighten 函数用于把图像增亮或变暗。其调用格式为：

（1）brighten(beta)

（2）brighten(h,beta)

（3）newmap = brighten(beta)

（4）newmap = brighten(cmap,beta)

如果 0<beta<1，即增亮图像；如果-1<beta<0，则变暗图像。

```
>> clear all;
figure('Renderer','zbuffer'); axesm bries
text(1.2, -1.8, 'Briesemeister projection')
framem('FLineWidth',1)
load topo
geoshow(topo, topolegend, 'DisplayType', 'texturemap')
demcmap(topo)
set(gcf,'color','w')
%增加图像的亮度
brighten(.5)
```

空域滤波原理

- 滤波器概念
  - 滤波器是一个大小为M×N的窗口，窗口中的元素与原矩阵的处于窗口内的元素进行某种运算，结果作为新矩阵的一个元素。当窗口滑过原矩阵并完成上述运算之后，就能够得到一个新矩阵。
  - 滤波器的别名：滤波器、掩模、核、模板，窗口

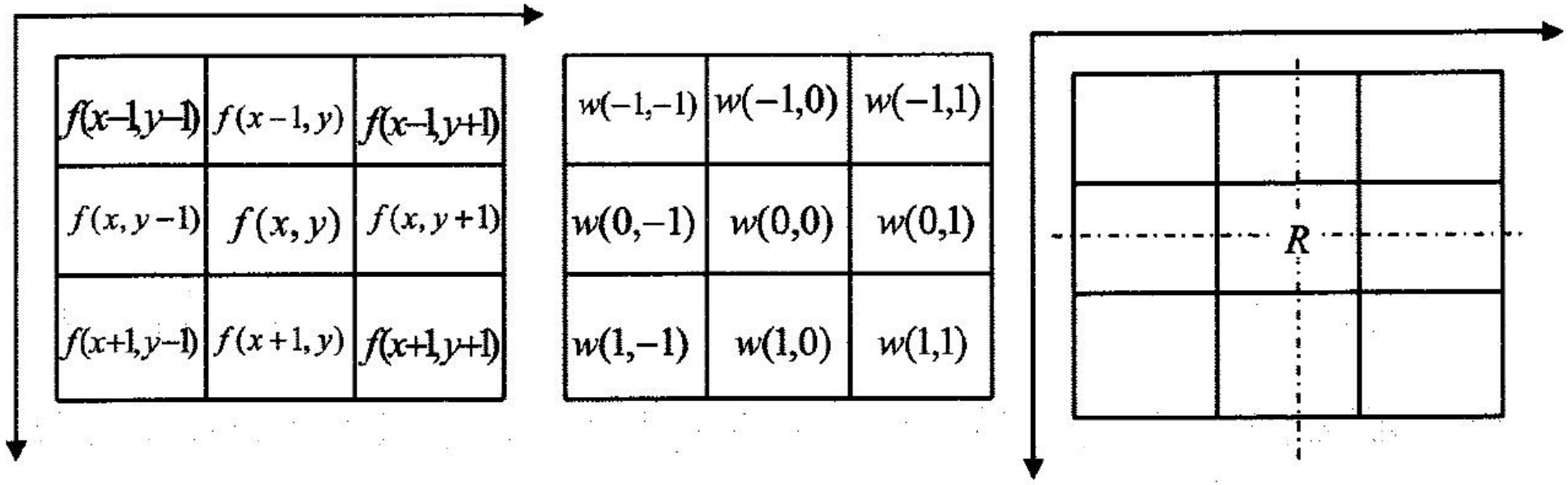
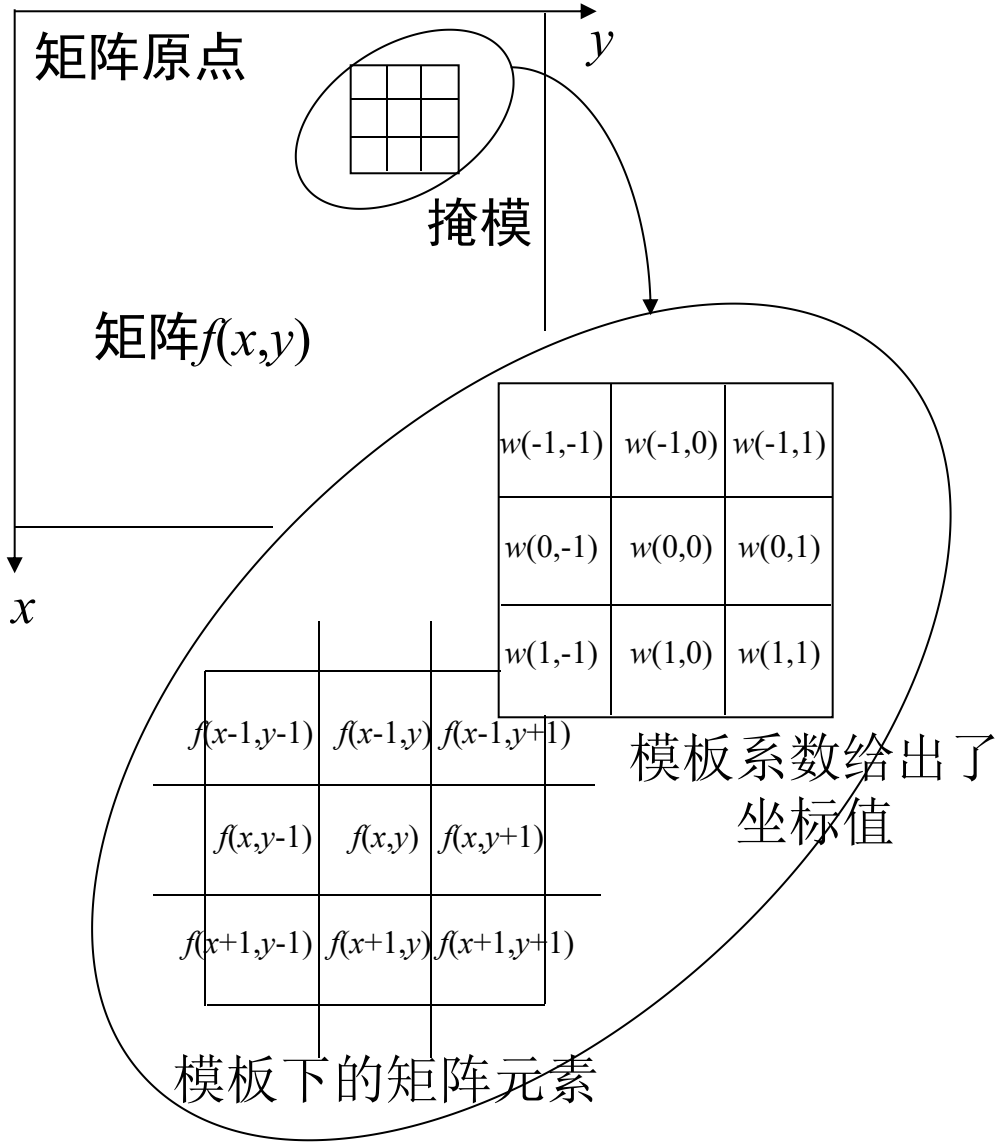


图 6-20　空域滤波原理

空域滤波原理

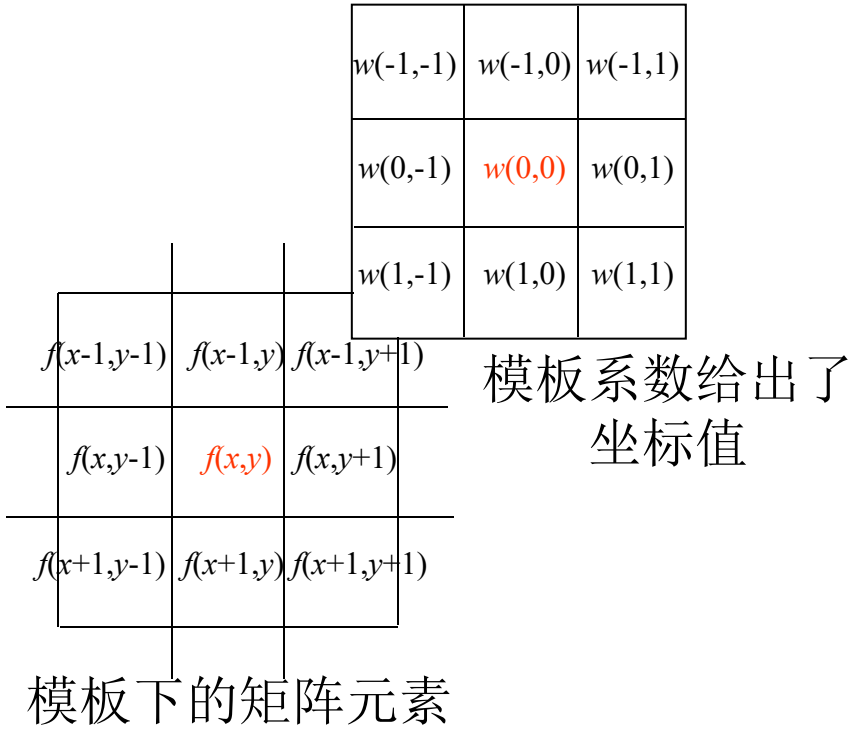
- 滤波过程

在待处理矩阵中逐点移动掩模，在每一点 (x, y) 处，滤波器在该点的响应通过实现定义的关系来计算。对于线性空间滤波，其响应由滤波器系数与滤波掩模扫过区域的对应元素值的乘积之和给出。



空域滤波原理

- 相应公式



$$R = w(-1,-1)f(x-1,y-1) + w(-1,0)f(x-1,y) + \cdots$$
$$+ w(0,0)f(x,y) + \cdots + w(1,0)f(x+1,y) + w(1,1)f(x+1,y+1)$$

空域滤波原理

- 响应公式
  - 通常，掩模的长宽都为奇数。假设分别为$2a+1$和$2b+1$。当窗口中心处于元素(x,y)处时，新的元素值为：

$$g(x,y)=\sum_{s=-a}^{a}\sum_{t=-b}^{b}w(s,t)f(x+s,y+t)$$

  - 简化形式：

$$R=w_1z_1+w_2z_2+\cdots+w_{mn}z_{mn}=\sum_{i=1}^{mn}w_iz_i$$

  - 对矩阵f中所有元素都与掩模进行运算之后，最终产生一个新矩阵g。

图像的空域滤波增强——平滑滤波

平滑运算的目的是消除或尽量减少噪声，改善图像的质量。假设加性噪声为随机独立分布，这样利用邻域的平均或加权平均就可以有效地抑制噪声干扰。从信号分析的观点来看，图像平滑本质上是低通滤波器，即通过信号的低频部分，阻截高频的噪声信号。但由于图像边缘也处于高频部分，这样往往带来另外一个问题——在对图像进行平滑操作时，往往对图像的细节造成一定程度的破坏。

如果 S 为像素 (x_0,y_0) 的邻域集合（包含 (x_0,y_0)），(x,y) 表示 S 中的元素，$f(x,y)$ 表示 (x,y) 点的灰度值，$a(x,y)$ 表示各点的权重，则对 (x_0,y_0) 进行平滑可表示为：

$$f'(x_0,y_0)=\frac{1}{\sum_{(x,y)\in S}a(x,y)}\left[\sum_{(x,y)\in S}a(x,y)f(x,y)\right]$$

一般而言，权重相对中心都是对称的。对于如下 3×3 大小的模板，其权重都是相等的：

$$T=\frac{1}{5}\begin{bmatrix}0&1&0\\1&1&1\\0&1&0\end{bmatrix}$$

对于这个模板，对应的函数表达式为：

$$f'(x,y)=\frac{1}{5}[f(x,y-1)+f(x-1,y)+f(x,y)+f(x+1,y)+f(x,y+1)]$$

它可表示为：

$$f'(x,y)=\frac{1}{5}[1\times f(x,y-1)+1\times f(x-1,y)+\cdots+1\times f(x,y+1)]=\frac{1}{5}T\times f$$

也就是说，邻域运算可以用邻域与模板的卷积得到，这也极大地方便了计算。

空域平滑滤波

- 图像在传输过程中，由于传输信道、采样系统质量较差，或受各种干扰的影响，而造成图像毛糙，此时，就需对图像进行平滑处理。平滑可以抑制高频成分，但也使图像变得模糊。
- 平滑空间滤波器用于模糊处理和减少噪声。模糊处理经常用于预处理，例如，在提取大的目标之前去除图像中一些琐碎的细节，桥接直线或曲线的缝隙。

经过低通滤波处理后的图像

线性平滑滤波器

- 概念
  - 平滑线性空间滤波器的输出是包含在滤波掩模邻域内元素的简单平均值。因此，这些滤波器也称为均值滤波器。
  - 简单平均，表示窗口中每一个元素对响应的贡献是一样的。
  - 加权平均，表示窗口中的元素对相应的贡献有大小之分。

$$\frac{1}{9}\times\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

$$\frac{1}{16}\times\begin{bmatrix}1&2&1\\2&4&2\\1&2&1\end{bmatrix}$$

线性平滑滤波器

- 一般公式

$$g(x,y)=\frac{\sum_{s=-a}^{a}\sum_{t=-b}^{b}w(s,t)f(x+s,y+t)}{\sum_{s=-a}^{a}\sum_{t=-b}^{b}w(s,t)}$$

其中，滤波器大小为$(2a+1)\times(2b+1)$，w为滤波器，f为输入矩阵，g为输出矩阵。

如果用于图像，滤波掩模的大小与图像的平滑效果有直接的关系。当掩模比较小时，可以观察到在整幅图像中有轻微的模糊，当掩模大小增加，模糊程度也随之增加。

非线性滤波器

- 中值滤波
  - 统计滤波器是一种非线性的空间滤波器，它的响应是基于窗口内图像区域中像素值的排序，由统计排序结果决定的值代替中心像素的值。
  - 统计滤波器中最常见的例子就是中值滤波器。

  - ➢ 用像素邻域内灰度的中值代替该像素的值。
  - ➢ 提供了优秀的去噪能力，比小尺寸的线性平滑滤波器的模糊程度明显要低。
  - ➢ 对处理脉冲噪声（也称为椒盐噪声）非常有效，因为这种噪声是以黑白点叠加在图像上的。

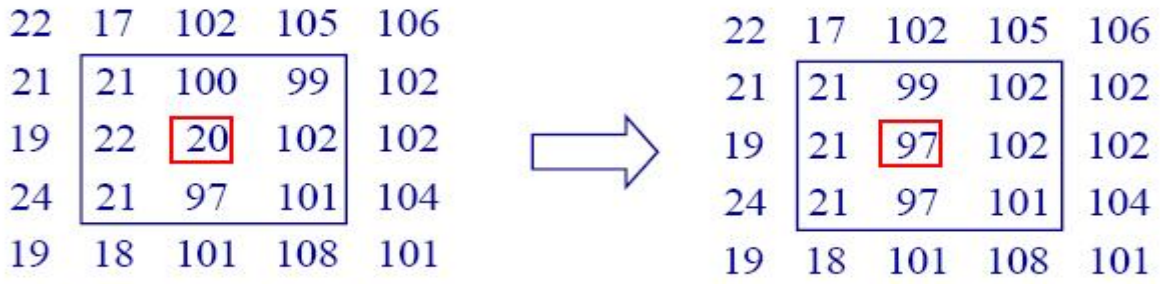
中值滤波器

- 中值计算
  - 中值ξ——数值集合中，有一半数值小于或等于ξ，还有一半大于或等于ξ。
  - 为了对一幅图像上的某个点作中值滤波处理。必须先将掩模内欲求的像素及其邻域的像素值排序，确定出中值，并将中值赋予该像素点。

在一个3×3的邻域内有一系列像素值：
（21，100，99，22，20，102，97，101）

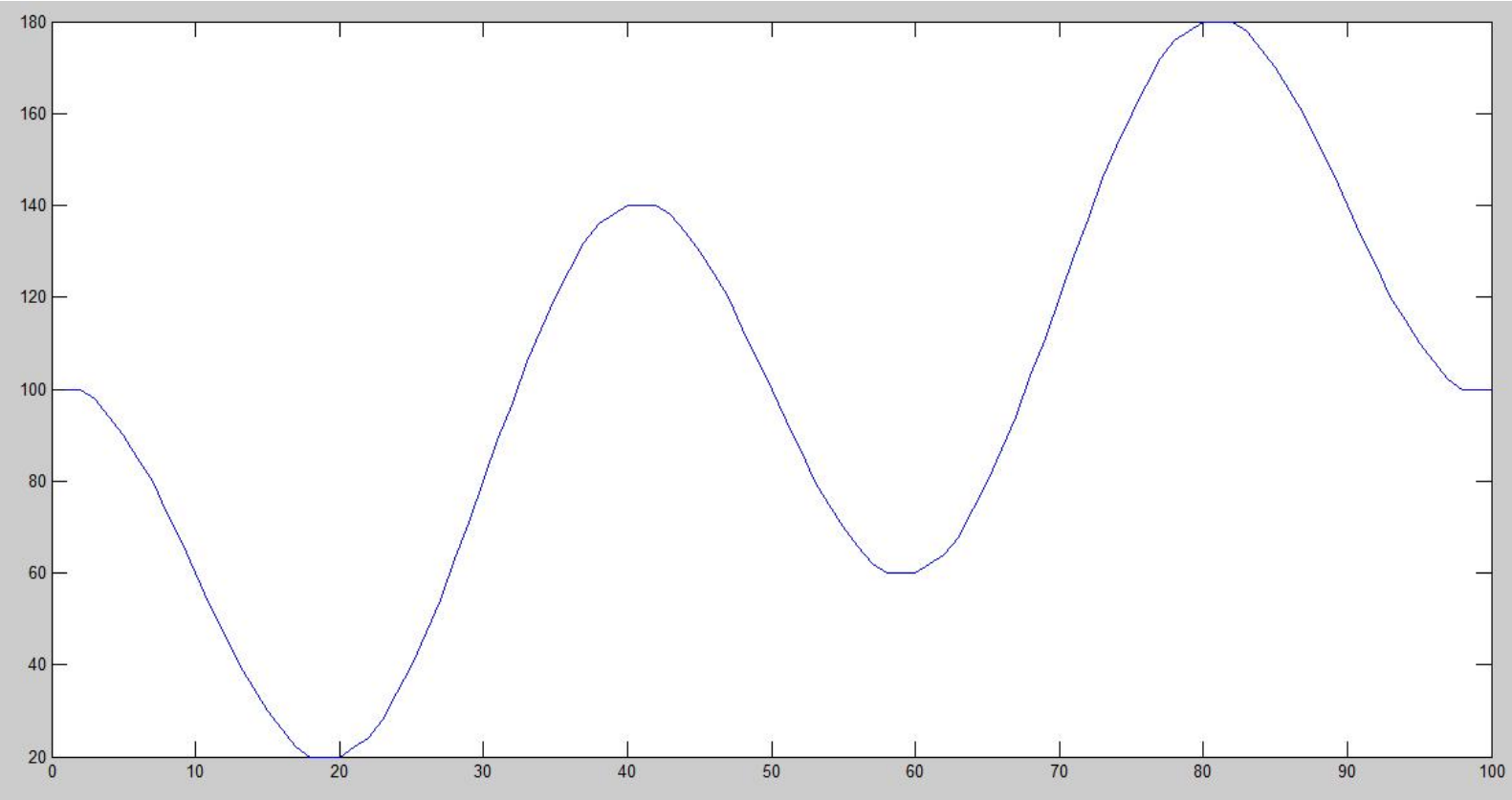
对这些值排序后为：
（20，21，22，97，100，101，012）

中值：97



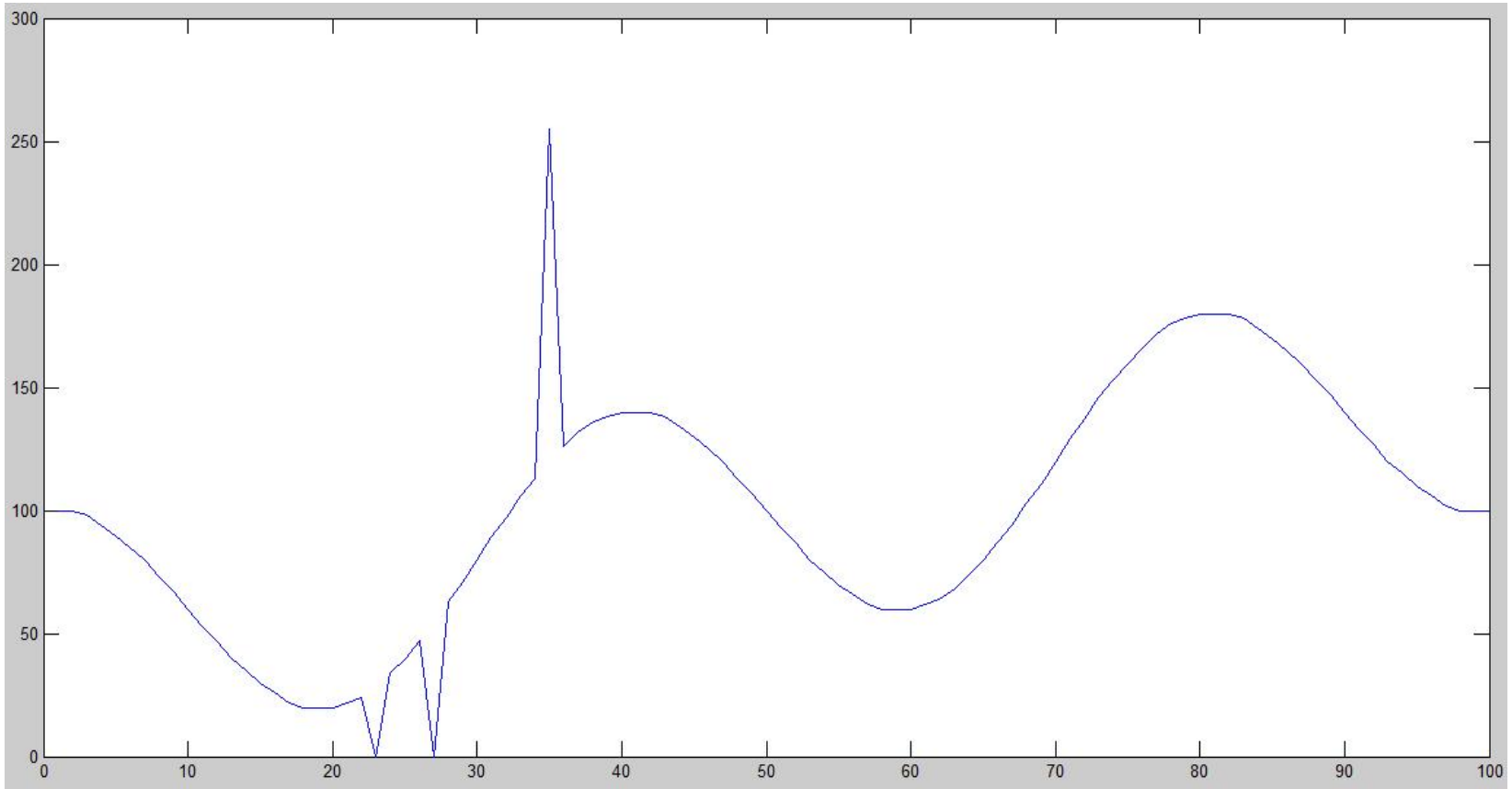
平滑滤波实例

- 实例：一维信号去噪
  - x = (1:100) + 50*cos((1:100)*2*pi/40)+50;



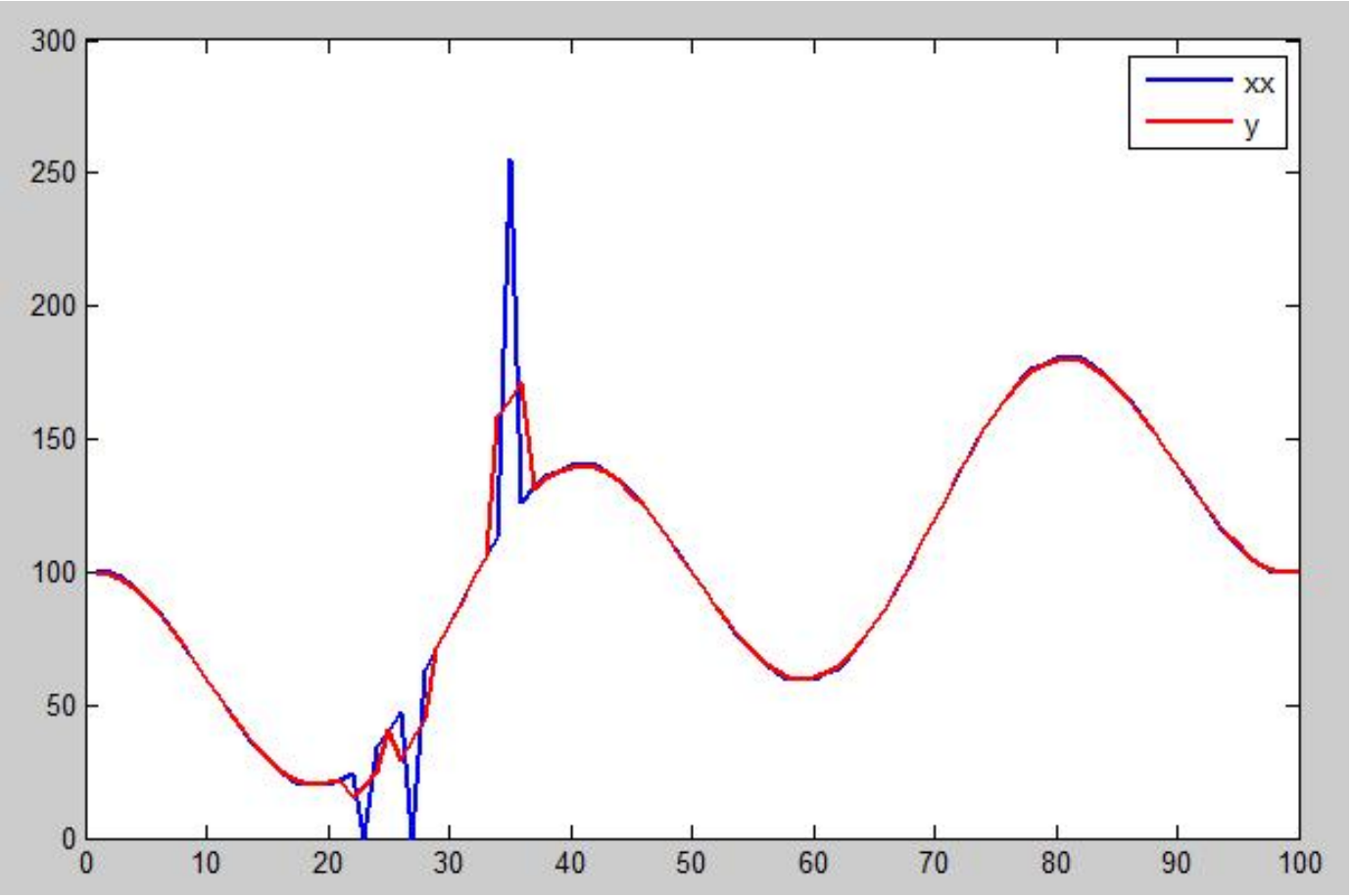
平滑滤波实例

- xx=imnoise(x,'salt & pepper',0.04);



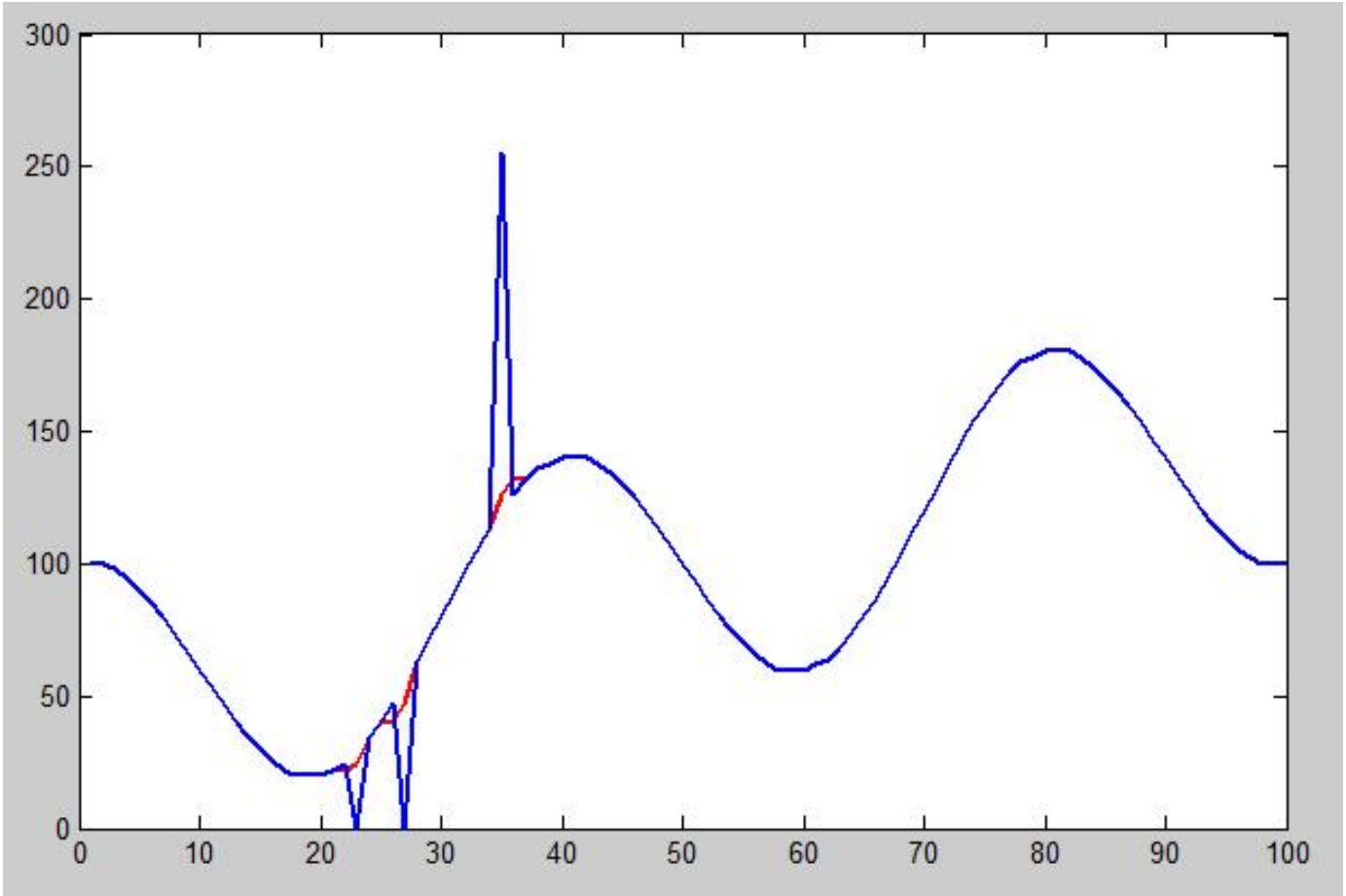
平滑滤波实例

- 均值滤波结果：



平滑滤波实例

- 中值滤波结果：



平滑滤波实例

- 实例：二维信号去噪（图像）

平滑滤波实例

- imn = imnoise(im,'salt & pepper',0.04);

平滑滤波实例

- I_Filter1 = medfilt2(imn,[3 3]); （中值滤波）

平滑滤波实例

- 均值滤波结果

线性平滑滤波

```
I= imread('peppers.png');                    % 读入图像
subplot(2,3,1),imshow(I);                    % 显示原始图像
title('原始图像');
I=rgb2gray(I);
J=imnoise(I,'salt & pepper',0.03);           % 加均值为 0，方差为 0.03 的椒盐噪声
subplot(2,3,2),imshow(J);                    % 显示处理后的图像
title('椒盐噪声');                                                  % 设置图像标题
K = filter2(fspecial('average',3),J)/255;
subplot(2,3,3),imshow(K,[]);
title('椒盐噪声被滤波后的图像');
J2=imnoise(I,'gaussian',0.03);               % 加均值为 0，方差为 0.03 的高斯噪声
subplot(2,3,4),imshow(J2);
title('高斯噪声');                                                  % 设置图像标题
K2 = medfilt2(J2);                           % 图像滤波处理
subplot(2,3,5),imshow(K2,[]);
title('高斯噪声被滤波后的图像');
```

线性平滑滤波

```
I= imread('peppers.png');                          % 读入图像
subplot(2,3,1),imshow(I);                          % 显示原始图像
title('原始图像');
I=rgb2gray(I);
J=imnoise(I,'salt & pepper',0.03);                 % 加均值为 0，方差为 0.03 的椒盐噪声
subplot(2,3,2),imshow(J);                          % 显示处理后的图像
title('椒盐噪声');                                                    % 设置图像标题
K = filter2(fspecial('average',5),J)/255;
subplot(2,3,3),imshow(K,[]);
title('椒盐噪声被5*5的模板滤波后的图像');
J2=imnoise(I,'gaussian',0.03);                                  % 加均值为 0，方差为 0.03 的高斯噪声
subplot(2,3,4),imshow(J2);
title('高斯噪声');                                                    % 设置图像标题
K2 = medfilt2(J2);                          % 图像滤波处理
subplot(2,3,5),imshow(K2,[]);
title('高斯噪声被5*5的模板滤波后的图像');
```

线性平滑滤波

```
I=imread('peppers.png');
subplot(231)
imshow(I)
title('原始图像')
I=rgb2gray(I);
I1=imnoise(I,'salt & pepper',0.02);
subplot(232)
imshow(I1)
title('添加椒盐噪声的图像')
k1=filter2(fspecial('average',3),I1)/255;        %进行3*3模板平滑滤波
k2=filter2(fspecial('average',5),I1)/255;        %进行5*5模板平滑滤波
k4=filter2(fspecial('average',9),I1)/255;        %进行9*9模板平滑滤波
k3=filter2(fspecial('average',7),I1)/255;        %进行7*7模板平滑滤波
subplot(233),
imshow(k1);
title('3*3模板平滑滤波');
subplot(234),
imshow(k2);
title('5*5模板平滑滤波');
subplot(235),
imshow(k3);
title('7*7模板平滑滤波');
subplot(236),
imshow(k4);
title('9*9模板平滑滤波');
```

中值滤波

```
I=imread('eight.tif');                % 读入图像
subplot(2,3,1),imshow(I);             % 显示原始图像
title('原始图像');
J=imnoise(I,'salt & pepper',0.03);    % 加均值为 0，方差为 0.03 的椒盐噪声
subplot(2,3,2),imshow(J);             % 显示处理后的图像
title('椒盐噪声');   % 设置图像标题
K = medfilt2(J);
subplot(2,3,3),imshow(K,[]);
title('椒盐噪声被滤波后的图像');
J2=imnoise(I,'gaussian',0.03);        % 加均值为 0，方差为 0.03 的高斯噪声
subplot(2,3,4),imshow(J2);
title('高斯噪声');        % 设置图像标题
K2 = medfilt2(J2);                    % 图像滤波处理
subplot(2,3,5),imshow(K2,[]);
title('高斯噪声被滤波后的图像');
```

中值滤波

```
I=imread('eight.tif');                % 读入图像
subplot(2,3,1),imshow(I);             % 显示原始图像
title('原始图像');
J=imnoise(I,'salt & pepper',0.03);    % 加均值为 0，方差为 0.03 的椒盐噪声
subplot(2,3,2),imshow(J);             % 显示处理后的图像
title('椒盐噪声');   % 设置图像标题
K = medfilt2(J,[5,5]);
subplot(2,3,3),imshow(K,[]);
title('椒盐噪声被被5*5的滤波后的图像');
J2=imnoise(I,'gaussian',0.03);        % 加均值为 0，方差为 0.03 的高斯噪声
subplot(2,3,4),imshow(J2);
title('高斯噪声');        % 设置图像标题
K2 = medfilt2(J2,[5,5]);              % 图像滤波处理
subplot(2,3,5),imshow(K2,[]);
title('高斯噪声被被5*5的滤波后的图像');
```

中值滤波

```
I=imread('eight.tif');
J=imnoise(I,'salt & pepper',0.03);
subplot(231),
imshow(I);
title('原图像');
subplot(232),
imshow(J);
title('添加椒盐噪声图像');
k1=medfilt2(J);              %进行3*3模板中值滤波
k2=medfilt2(J,[5,5]);        %进行5*5模板中值滤波
k3=medfilt2(J,[7,7]);        %进行7*7模板中值滤波
k4=medfilt2(J,[9,9]);        %进行9*9模板中值滤波
subplot(233),
imshow(k1);
title('3*3模板中值滤波');
subplot(234),
imshow(k2);
title('5*5模板中值滤波');
subplot(235),
imshow(k3);
title('7*7模板中值滤波');
subplot(236),
imshow(k4);
title('9*9模板中值滤波');
```

锐化空间滤波器

- 目的
  - 突出图像中的细节或者增强被模糊了的细节。

- 工具
  - 微分算子是实现锐化的工具，其响应程度与图像在该点处的突变程度有关。微分算子增强了边缘和其他突变（如噪声）并削弱了灰度变化缓慢的区域。

锐化空间滤波器

- 微分定义
  - 对于函数$f(x)$，用差值来表达一阶微分：

$$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$

  - 类似地，用差分定义二阶微分：

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x)$$

拉普拉斯算子

- 基于二阶微分的图像增强
  - 拉普拉斯算子是一种各向同性滤波器，其响应和图像突变方向无关。各向同性滤波器是旋转不变的，即将原始图像旋转后进行滤波处理得出的结果与先对图像滤波，然后再旋转的结果相同。
  - 公式：

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

  - 离散形式：

$$\frac{\partial^2 f}{\partial x^2} = f(x+1,y) + f(x-1,y) - 2f(x,y)$$

$$\frac{\partial^2 f}{\partial y^2} = f(x,y+1) + f(x,y-1) - 2f(x,y)$$

$$\nabla^2 f = [f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1)] - 4f(x,y)$$

拉普拉斯算子

- 掩模

0	1	0
1	-4	1
0	1	0

- 对角线方向也可以加入到拉普拉斯变换的定义中，其定义为：

$$\begin{aligned}\nabla^2 f = [&f(x-1,y-1)+f(x,y-1)+f(x+1,y-1)\\&+f(x-1,y)+f(x+1,y)\\&+f(x-1,y+1)+f(x,y+1)+f(x+1,y+1)]\\&-8f(x,y)\end{aligned}$$

1	1	1
1	-8	1
1	1	1

拉普拉斯算子

- 其它扩展形式

0	-1	0
-1	4	-1
0	-1	0

-1	-1	-1
-1	8	-1
-1	-1	-1

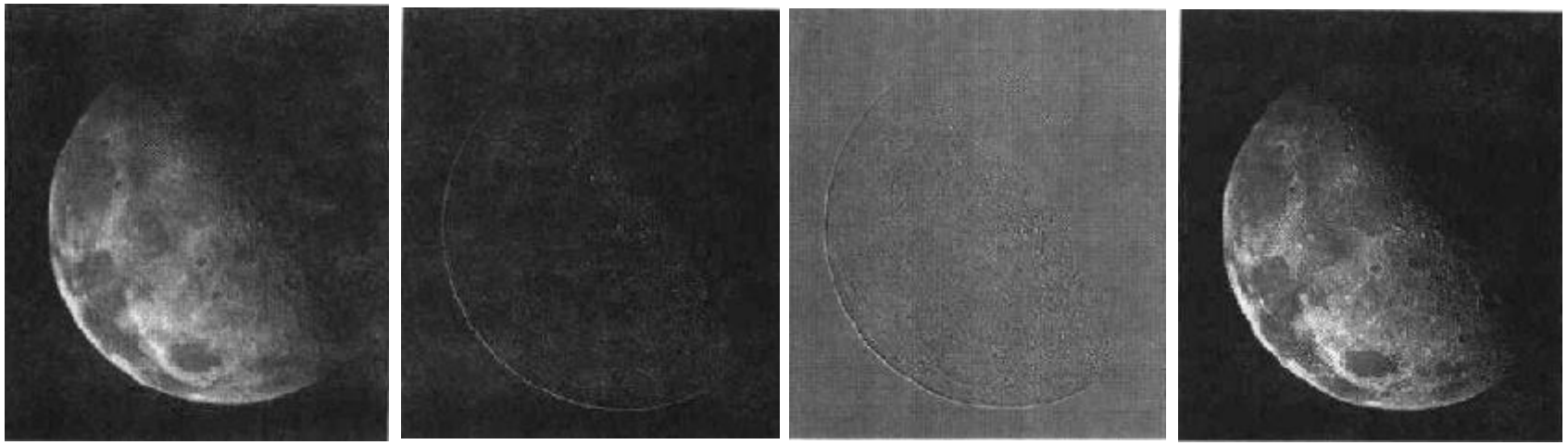
与前述掩模符号相反。当拉普拉斯滤波后的图像与其它图像合并时（相加或相减），则必须考虑符号上的差别。

拉普拉斯算子

- 应用
  - 使用拉普拉斯变换对图像增强的基本方法可表示为下式：

$$g(x,y)=\begin{cases} f(x,y)-\nabla^2 f(x,y) & \text{如果拉普拉斯掩模中心系数为负} \\ f(x,y)+\nabla^2 f(x,y) & \text{如果拉普拉斯掩模中心系数为正} \end{cases}$$

将原始图像和拉普拉斯图像叠加在一起的简单方法可以保护拉普拉斯锐化处理的效果，同时又能复原背景信息。

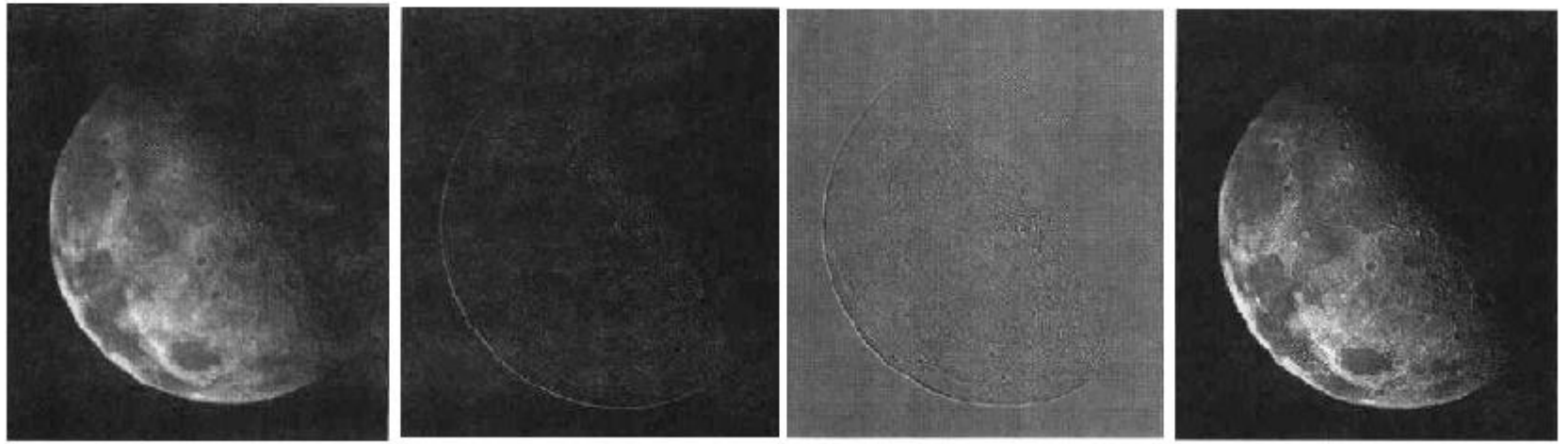


(a) 原始图像　(b) 拉普拉斯图像　(c) 归约化后的拉普拉斯图像　(d) 增强了的图像

- 使用拉普拉斯变换对图像增强的基本方法可表示为下式：

$$g(x,y)=\begin{cases} f(x,y)-\nabla^2 f(x,y) & \text{如果拉普拉斯掩模中心系数为负} \\ f(x,y)+\nabla^2 f(x,y) & \text{如果拉普拉斯掩模中心系数为正} \end{cases}$$

将原始图像和拉普拉斯图像叠加在一起的简单方法可以保护拉普拉斯锐化处理的效果，同时又能复原背景信息。



(a)原始图像 (b)拉普拉斯图像 (c)归约化后的拉普拉斯图像 (d)增强了的图像


```
f = imread('1.jpg')
f_lap  = LaplaceFilter(f);
g = f + f_lap
```

```
imshow(uint8(abs(f_lap)));
```

```
imshow(f_lap,[]);
```

(b)：相当于画出f_lap的幅值

(c)： 规约化是指：把最小值映射到0，最大值映射到255

拉普拉斯算子

- 步骤1：

(a) 原始图像

1	1	1
1	−8	1
1	1	1

(b) 拉普拉斯图像

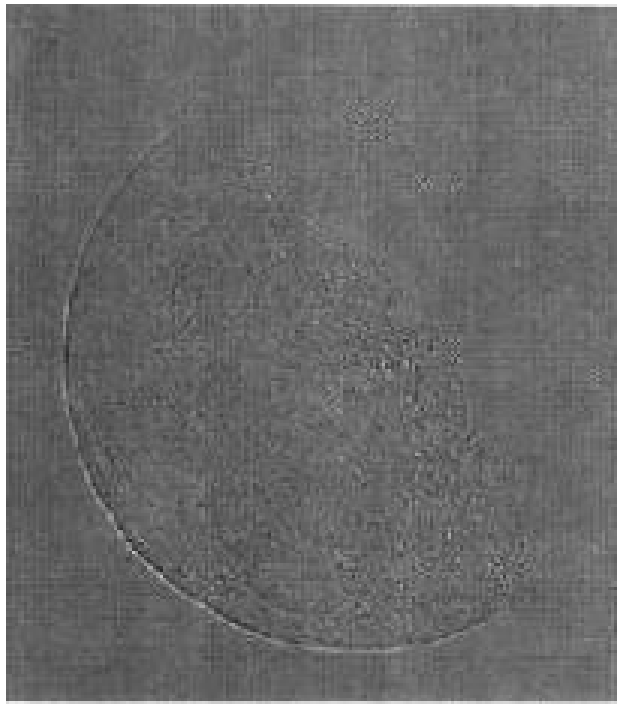
图 (b) 展示的 $\mathrm{abs}(\nabla^2 f(x, y))$

拉普拉斯算子

- 步骤2：

(b) 拉普拉斯图像

处理拉普拉斯图像中的像素值，使其限定在有效范围之内。



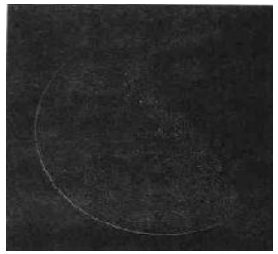
(c) 归约化后的拉普拉斯图像

图（b)展示的 abs(▽²f(x, y))

实际操作中，是把▽²f(x, y) 规约化到0~255；而非abs(▽²f(x, y))

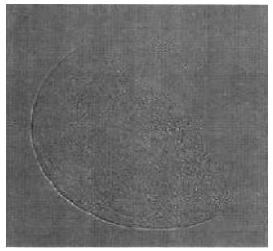
拉普拉斯算子

- 步骤3：



(b)

$\mathrm{abs}(\nabla^2 f(x,y))$



(c)

规约化到0~255之后的
$\nabla^2 f(x,y)$

(a) 原始图像

$\nabla^2 f(x,y)$

叠加

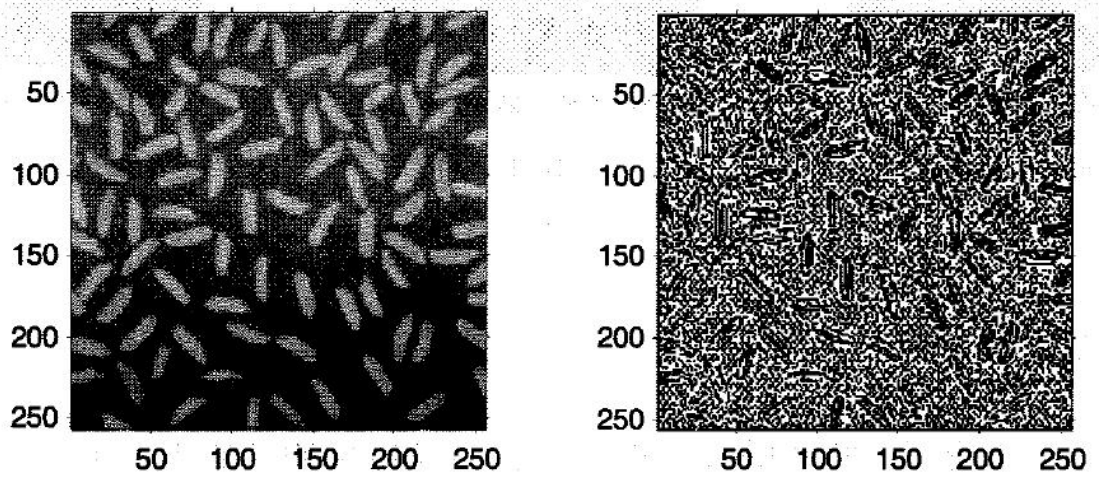
(d) 增强了的图像

拉普拉斯算子

【例 6-21】对图像进行线性锐化滤波。

```
>> clear all;
I=imread('rice.png');
h=fspecial('laplacian');                    %得到用于滤波的滤波器
I2=filter2(h,I);
subplot(121);imshow(I);
xlabel('(a)原始图像');
subplot(122);imshow(I2);
xlabel('(b)滤波后图像')
```

运行程序，结果如图 6-28 所示。



（a）原始图像　　（b）滤波后图像

课程实验5

- 编写代码实现：
  - 编写函数实现基于拉普拉斯算子的图像增强，输入为原始图像和某个自定义的拉普拉斯算子，输出为增强后的图像