第6章 图像平滑

主讲教师　张涛

电子信息与电气工程学院

问题的提出：

- 一幅图像，常常不仅包括我们感兴趣的目标物体，同时也包含一些噪声或干扰，这是由于在图像的获取、传输和存储过程中，受到各种噪声的干扰和影响，使图像质量下降。
- 如何能够衰减或去除噪声？

问题的提出：

- 图像平滑（**Image Smoothing**）：通常指抑制或消除图像中存在的噪声而改善图像质量的过程。
- 图像平滑方法分类
  - 空域法：主要借助模板运算，在像素点邻域内，利用噪声像素点特性进行滤波；
  - 频域法：指对图像进行正交变换，利用噪声对应高频信息的特点进行滤波。

主要内容

6.1 图像中的噪声

6.2 空间域平滑滤波

6.3 频域平滑滤波

6.4 其他图像平滑滤波

6.1 图像中的噪声

- 所谓噪声，可以理解为“妨碍人的视觉器官或系统传感器对所接收的图像信息进行理解或分析的各种因素”，也可以理解为“真实信号与理想信号之间存在的偏差”。
- 噪声通常以二维函数$n(x, y)$来表示。

6.1 图像中的噪声

6.1.1 图像噪声的分类

6.1.2 图像噪声的数学模型

6.1.1 图像噪声的分类

- 图像噪声主要可能来源
  - 光电传感器噪声；
  - 大气层电磁暴、闪电等引起的强脉冲干扰；
  - 相片颗粒噪声；
  - 信道传输误差引起的噪声等。

6.1.1 图像噪声的分类

- 图像噪声可根据噪声源
  - 高斯噪声
  - 泊松噪声
  - 颗粒噪声

6.1.1 图像噪声的分类

- 图像噪声可根据信号和噪声的关系
  - 加性噪声——加性噪声与图像信号不相关

$$g(x,y)=f(x,y)+n(x,y)$$

  - 乘性噪声——乘性噪声与图像信号相关

$$g(x,y)=f(x,y)+f(x,y)n(x,y)$$

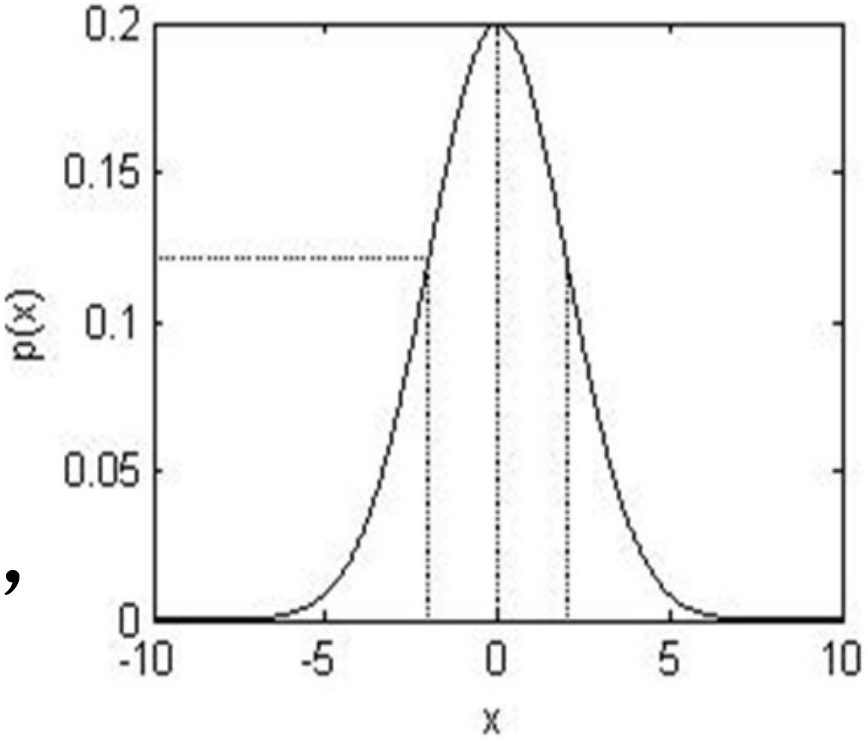
在信号变化很小时，往往将乘性噪声近似认为加性噪声，而且总是假定信号和噪声互相独立。

6.1.2 图像噪声的数学模型

（1） 高斯噪声

定义为高斯噪声信号x的概率密度函数

$$p(x)=\frac{1}{\sqrt{2\pi}\sigma}e^{-(x-\mu)^2/2\sigma^2}$$



μ表示噪声x均值或期望值，
σ表示噪声x的标准差。

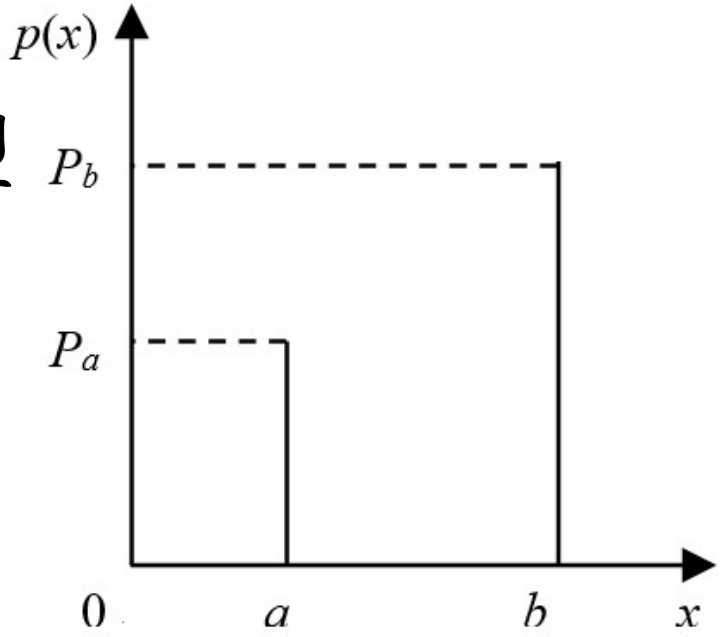
当x服从高斯分布时，其值有70%落在$[(\mu-\sigma),(\mu+\sigma)]$范围内，且有95%落在$[(\mu-2\sigma),(\mu+2\sigma)]$范围内。

6.1.2 图像噪声的数学模型

（2） 椒盐噪声

定义椒盐噪声信号x有效模型

$$p(x)=\begin{cases}P_a & x=a\\ P_b & x=b\\ 0 & otherwise\end{cases}$$



- 当$P_a=0, P_b\neq 0$时，表现为“盐”噪声。
- 当$P_a\neq 0, P_b=0$时，表现为“胡椒”噪声。
- 这都为孤立噪声点

6.1.2 图像噪声的数学模型

可看出，高斯噪声和椒盐噪声不同分布特性

- 椒盐噪声：噪声幅值基本相同，出现位置随机。
- 高斯噪声：噪声出现位置是分布在每一像素点上，幅度值是随机的，分布近似符合高斯正态特性。

6.1.2 图像噪声的数学模型

（3） 例程

- 函数

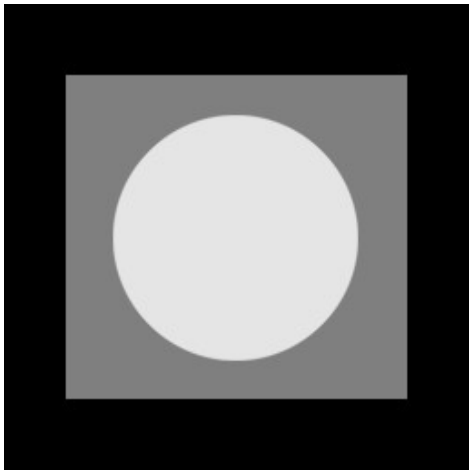
  J= imnoise (I,TYPE,PARAMETERS)

- 程序

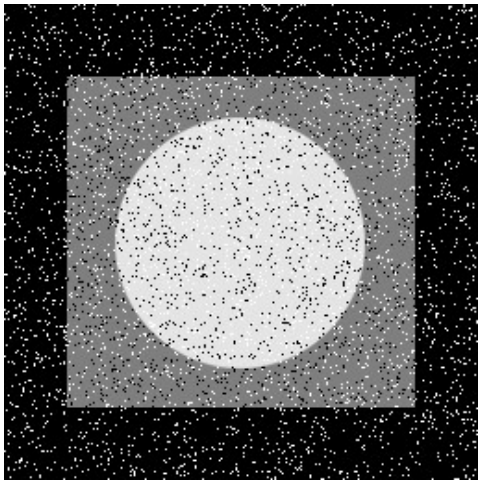
```
Image=mat2gray(imread('original_pattern.jpg')
,[0 255]);
noiseIsp=imnoise(Image,'salt&pepper',0.1);
imshow(noiseIsp,[0 1]); title('椒盐噪声');
noiseIg=imnoise(Image,'gaussian');
figure;imshow(noiseIg,[0 1]); title('高斯噪声');
```

6.1.2 图像噪声的数学模型

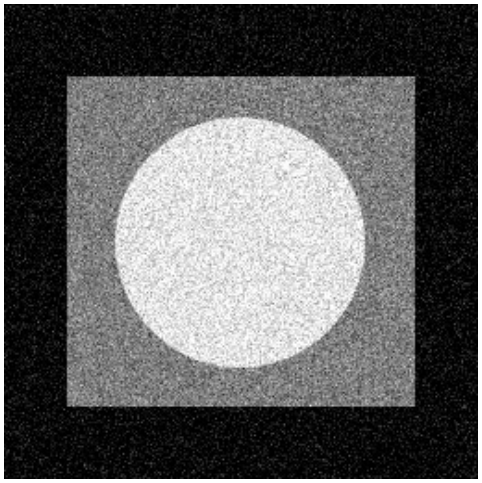
- 效果



（a）原始图像



（b） 椒盐噪声图像



（c）高斯噪声图像

6.2 空间域平滑滤波

空域滤波主要指的是基于图像空间的邻域模板运算。

6.2.1 均值滤波

6.2.2 高斯滤波

6.2.3 中值滤波

6.2.4 双边滤波

6.2.1 均值滤波

（1）均值滤波原理

是以某一像素为中心，在它的周围选择一邻域，将邻域内所有点的均值（灰度值相加求平均）来代替原来像素值。

$$g(x,y)=\frac{1}{M}\sum_{(m,n)\in S} f(m,n)$$

S：点(x,y)为中心的邻域；

M：邻域S内总像素数目

6.2.1 均值滤波

- 常用的线性平滑简单均值模板

$$H_1 = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad H_3 = \frac{1}{5}\begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

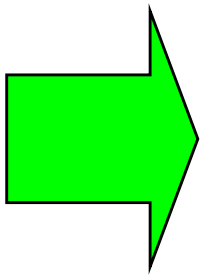
$$H_2 = \frac{1}{25}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \qquad H_4 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

6.2.1 均值滤波

（2）均值滤波示例

强调注意：边界像素处理？？

1	2	1	4	3
1	2	2	3	4
5	7	6	8	9
5	7	6	8	8
5	6	7	8	9



1	2	1	4	3
1	3	4	4	4
5	5	6	6	9
5	6	7	8	8
5	6	7	8	9

（图中计算结果按四舍五入进行了调整）

6.2.1 均值滤波

（3）均值滤波效果分析

- 若邻域内存在噪声，经过平均，噪声幅度会大为降低。
- 点与点之间的灰度差值变小，边缘和细节处变得模糊。
- 邻域半径越大，图像模糊程度越厉害。

6.2.1 均值滤波

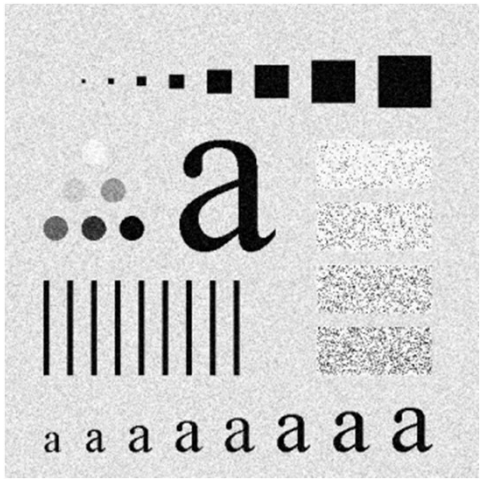
（4）例程

- 函数　H=fspecial (TYPE, PARAMETERS):
  Y = filter2(B,X,SHAPE):
  B= imfilter(A,H,OPTION1, OPTION2,…):
- 程序

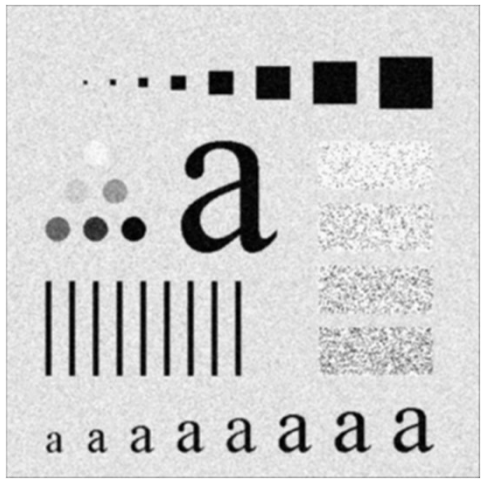
```
Image=imread('Letters-a.jpg');
noiseI=imnoise(Image,'gaussian');
result1=filter2(fspecial('average',3),noiseI);
result2=filter2(fspecial('average',7),noiseI);
subplot(223),imshow(uint8(result1)),title('3×3');
subplot(224),imshow(uint8(result2)),title('7×7');
```

6.2.1 均值滤波

- 效果



（a）高斯噪声图像



（b）3×3 均值滤波

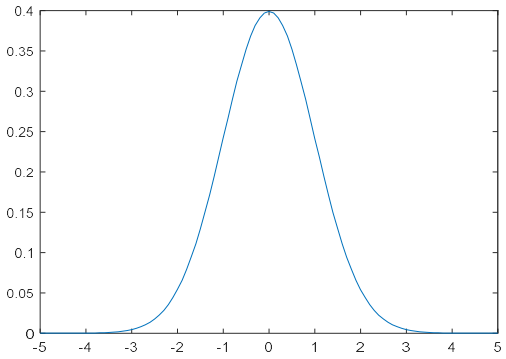
（c）7×7均值滤波

6.2.2 高斯滤波

（1）高斯函数

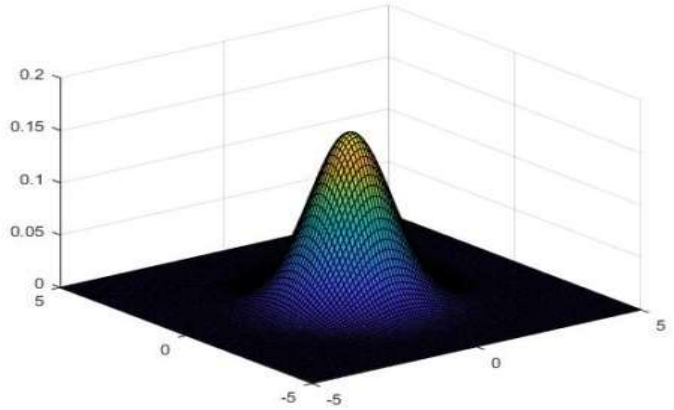
- 一维高斯函数

$$H(x)=\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{x^2}{2\sigma^2}}$$



- 二维高斯函数

$$H(x,y)=\frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}}$$



正态分布曲线为钟形形状；表明:离中心原点越近,高斯函数取值越大。正态分布常被用来进行权值分配。

6.2.2 高斯滤波

（2）高斯滤波原理

是以某一像素为中心，在它的周围选择一个局部邻域，把邻域内像素的灰度按照高斯正态分布曲线进行统计，分配相应的权值系数，然后将邻域内所有点的加权平均值代替原像素值。

$$g(x,y)=\sum_{r=-k}^{k}\sum_{s=-l}^{l}f(x-r,y-s)H(r,s)$$

其中，k,l是根据所选邻域大小确定。

6.2.2 高斯滤波

（3）高斯模板特点

- 按照正态分布曲线的统计，模板上不同位置赋予不同的加权系数值。
- 标准差σ影响高斯模板生成的关键参数。
  - σ值越大，生成高斯模板中不同系数值差别不大，类似均值模板，对图像的平滑效果较明显。

6.2.2 高斯滤波

- 典型的高斯模板
  - $\sigma = 0.8$:

$$H_1 = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

$$H_2 = \frac{1}{2070}\begin{bmatrix} 1 & 10 & 22 & 10 & 1 \\ 10 & 108 & 237 & 108 & 10 \\ 22 & 237 & 518 & 237 & 22 \\ 10 & 108 & 237 & 108 & 10 \\ 1 & 10 & 22 & 10 & 1 \end{bmatrix}$$

  - $\sigma = 1$:

$$H_3 = \frac{1}{10}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$$H_4 = \frac{1}{330}\begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 20 & 33 & 20 & 4 \\ 7 & 33 & 54 & 33 & 7 \\ 4 & 20 & 33 & 20 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix}$$

6.2.2 高斯滤波

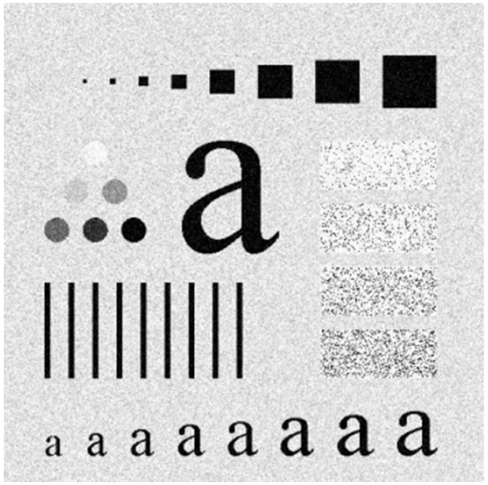
（4）例程

- 程序

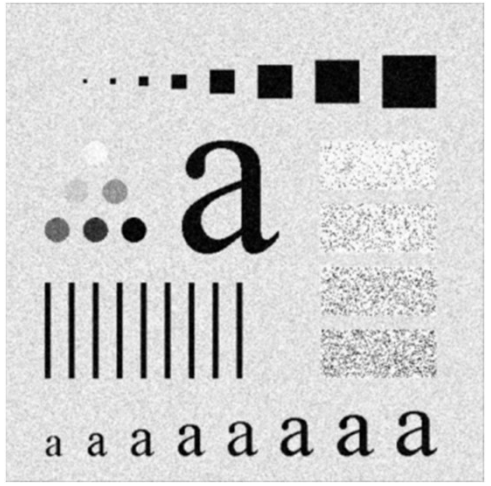
```
Image=imread('Letters-a.jpg');
sigma1=0.6;   sigma2=10;    r=3;
NoiseI= imnoise(Image,'gaussian');
gausFilter1=fspecial('gaussian',[2*r+1 2*r+1],sigma1);
gausFilter2=fspecial('gaussian',[2*r+1 2*r+1], sigma2);
result1=imfilter(NoiseI,gausFilter1,'conv');
result2=imfilter(NoiseI,gausFilter2,'conv');
figure;imshow(result1);title('sigma1=0.6高斯滤波');
figure;imshow(result2);title('sigma2=10高斯滤波');
```

6.2.2 高斯滤波

- 效果



（a）高斯噪声图像



（b）标准差=0.6的高斯滤波

（c）标准差=10的高斯滤波

6.2.3中值滤波

属于非线性平滑滤波。

（1）中值

一组数$x_1, x_2, \ldots, x_n$，把n个数按值的大小顺序排列如下：$x_{i1} < x_{i2} < \mathrm{L} < x_{in}$

中值

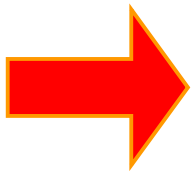
$$y = \begin{cases} x_{i(\frac{n+1}{2})} & n\text{为奇数} \\ \frac{1}{2}\left[x_{i(\frac{n}{2})} + x_{i(\frac{n}{2}+1)} \right] & n\text{为偶数} \end{cases}$$

6.2.3中值滤波

（2）中值滤波原理

- 噪声的出现，使被处理点像素比周围像素亮(暗)许多。
- 以被处理点为中心，选取一个邻域窗口，窗口内所有点值排序，取中值代替该点值。

212	200	198
206	202	201
208	205	207

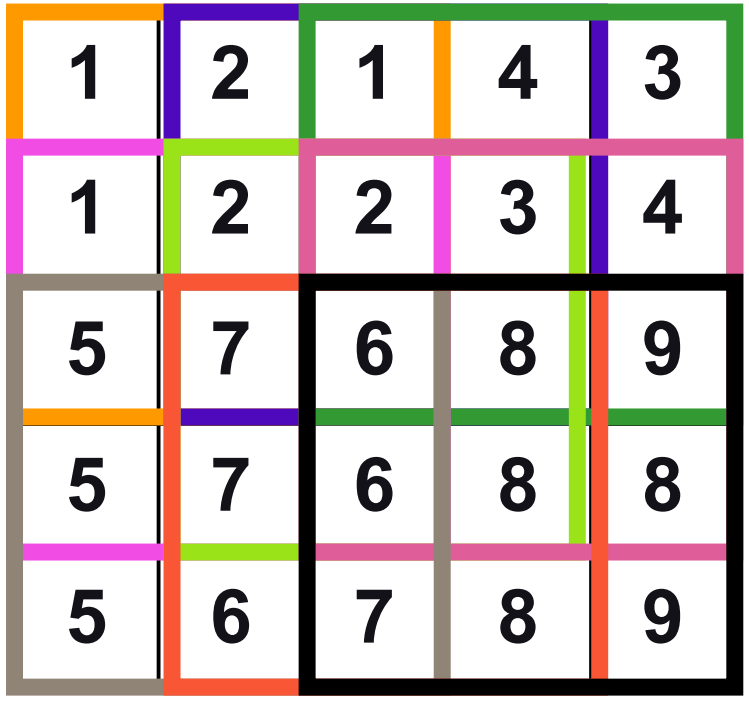


212	200	198
206	205	201
208	205	207

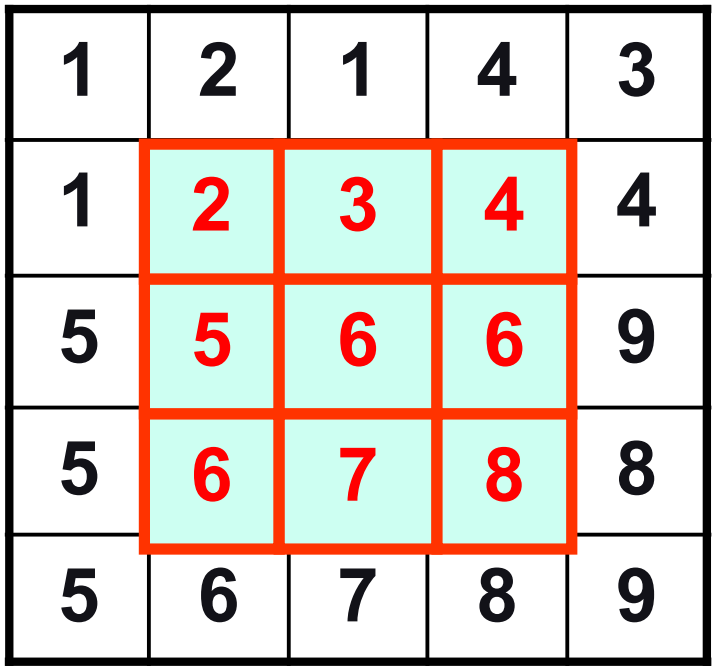
排序 198 200 201 202 205 206 207 208 212

6.2.3中值滤波

（3）示例---基于3×3邻域的中值滤波



原图像



中值滤波处理后输出图像

6.2.3中值滤波

（4）例程

- 函数

  B = medfilt2(A,[M N])

- 程序

```
Image=rgb2gray(imread('lotus.bmp'));
noiseI=imnoise(Image,'salt&pepper',0.1);
imshow(noiseI),title('椒盐噪声图像');
result=medfilt2(noiseI);
figure,imshow(uint8(result)),title('3×3中值滤波');
```

6.2.3中值滤波

- 效果

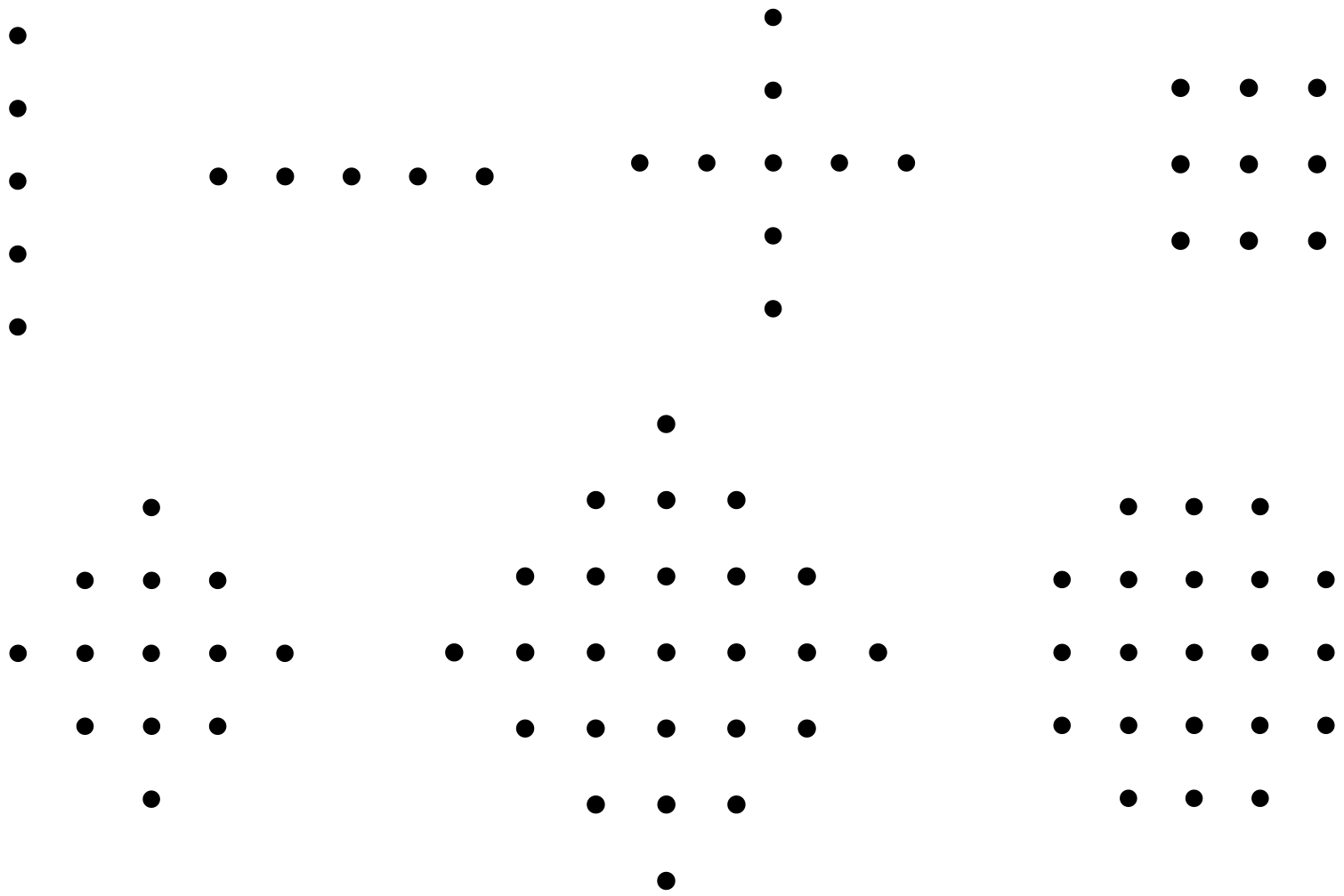
（a）原椒盐噪声图像

（b）3×3中值滤波效果

（c）3×3均值滤波效果

6.2.3中值滤波

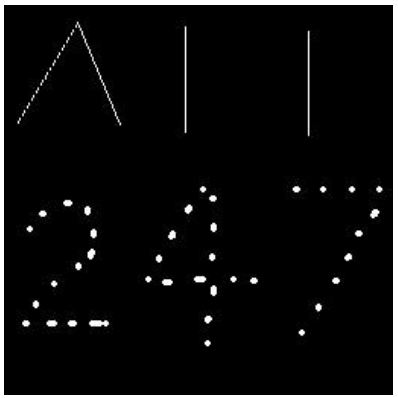
（5）中值滤波器形状



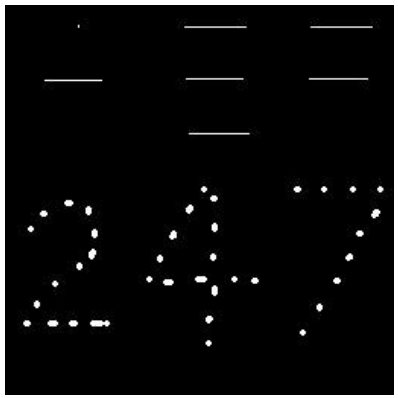
6.2.3中值滤波

- 不同形状中值滤波器效果

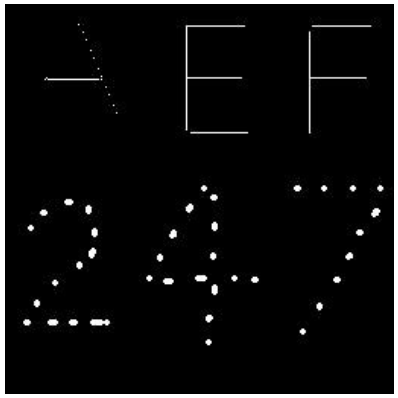
原图



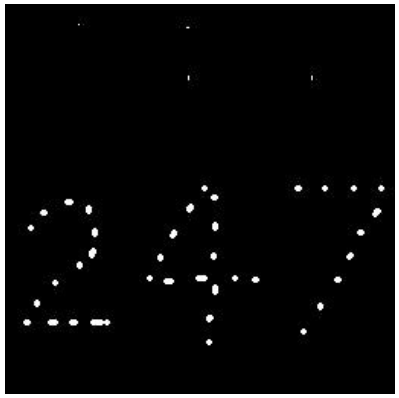
1×3滤波



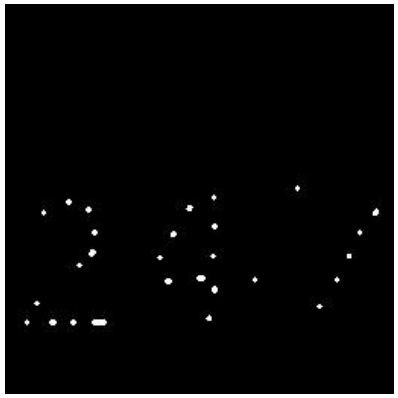
3×1滤波



4邻域滤波



8邻域滤波



5×5滤波

6.2.3中值滤波

（6）均值、中值滤波效果分析

椒盐噪声图像

均值滤波

4邻域

中值滤波

6.2.3中值滤波

（6）均值、中值滤波效果分析

椒盐噪声图像

均值滤波

3×3

中值滤波

6.2.3中值滤波

（6）均值、中值滤波效果分析

椒盐噪声图像

均值滤波

5×5

中值滤波

6.2.3中值滤波

（6）均值、中值滤波效果分析

- 与均值滤波相比，中值滤波去除椒盐噪声效果好，而且模糊轻微，边缘保留较好。
  - 椒盐噪声是幅值近似相等但随机分布在不同位置上，图像中有干净点也有污染点。
  - 中值滤波是选择适当的点来替代污染点的值，所以处理效果好。
  - 因为噪声的均值不为0，所以均值滤波不能很好地去除噪声点。

6.2.3中值滤波

（6）均值、中值滤波效果分析

- 对于高斯噪声，均值滤波效果比较好。
  - 高斯噪声是幅值近似正态分布，但分布在每点像素上。
  - 因为图像中的每点都是污染点，所中值滤波选不到合适的干净点。
  - 因为正态分布的均值为0，所以根据统计数学，均值可以消除噪声。
  - 实际上只能减弱，不能消除。

6.2.3中值滤波

（6）均值、中值滤波效果分析

- 中值滤波邻域窗口越大，图像模糊程度愈大。

6.2.4双边滤波

（1）原理

“双边”意味着平滑滤波时不仅考虑邻域内像素的空间邻近性，而且要考虑邻域内像素的灰度相似性。

$$BF[I]_P = \frac{1}{W_P}\sum_{q\in S} G_{\sigma_s}\left(\|p-q\|\right)G_{\sigma_r}\left(\left|I_p - I_q\right|\right)I_q$$

$$W_P = \sum_{q\in S} G_{\sigma_s}\left(\|p-q\|\right)G_{\sigma_r}\left(\left|I_p - I_q\right|\right)$$

$$G_{\sigma_s}\left(\|p-q\|\right) = e^{-\frac{\left(\|p-q\|\right)^2}{2\sigma_s^2}} \qquad G_{\sigma_r}\left(\left|I_p - I_q\right|\right) = e^{-\frac{\left(\left|I_p - I_q\right|\right)^2}{2\sigma_r^2}}$$

6.2.4双边滤波

（2）例程

- 程序

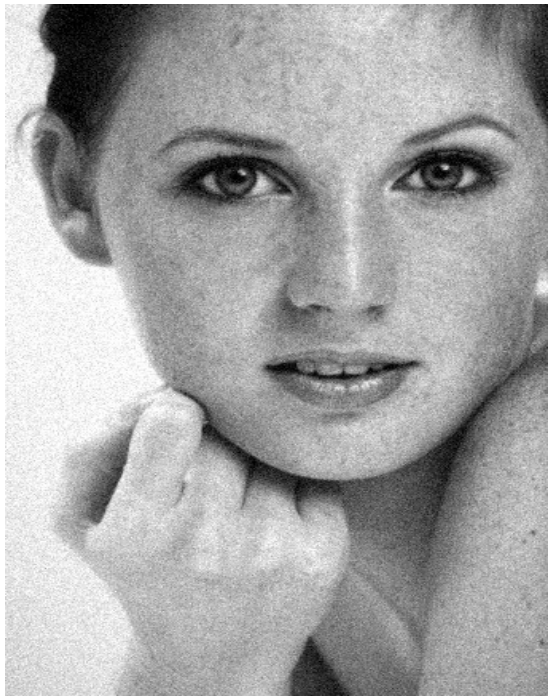
```
Image=im2double(imread('girl.bmp'));
NoiseI= Image+0.05*randn(size(Image));
w=15;   sigma_s=6;  sigma_r=0.1
[X,Y] = meshgrid(-w:w,-w:w);
Gs = exp(-(X.^2+Y.^2)/(2*sigma_s^2));
%计算邻域内的空间权值
[hm,wn] = size(NoiseI);
result=zeros(hm,wn);
```

6.2.4双边滤波

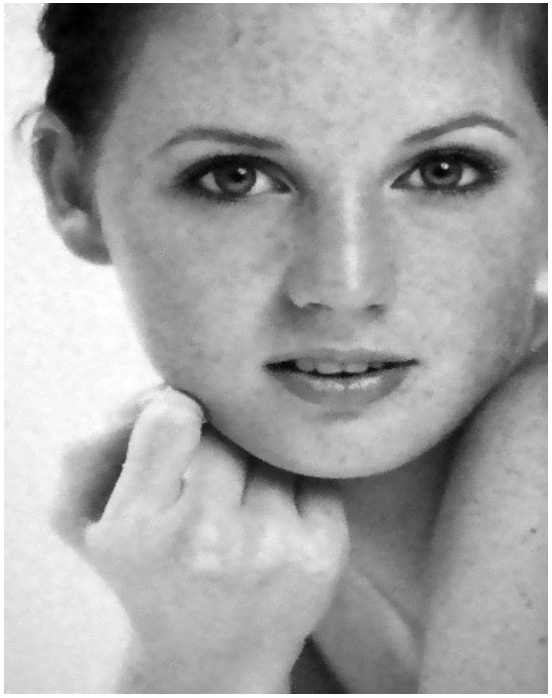

```
for i=1:hm
    for j=1:wn
        temp=NoiseI(max(i-w,1):min(i+w,hm),max(j-
w,1):min(j+w,wn));
        Gr = exp(-(temp-NoiseI(i,j)).^2/(2*sigma_r^2));
        W=Gr.*Gs((max(i-w,1):min(i+w,hm))-i+w+1,
                 (max(j-w,1):min(j+w,wn))-j+w+1);
        result(i,j)=sum(W(:).*temp(:))/sum(W(:));
    end
end
imshow(result),title('双边滤波图像');
```

6.2.4双边滤波

- 效果



（a）随机噪声图像



（b）$w=3, G\sigma_s=3, G\sigma_r=0.1$

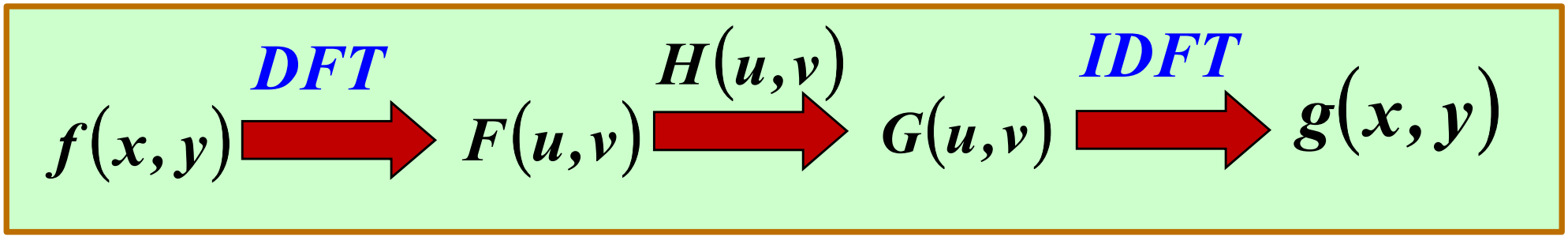
（c）$w=15, G\sigma_s=6, G\sigma_r=0.1$

6.3 频域平滑滤波

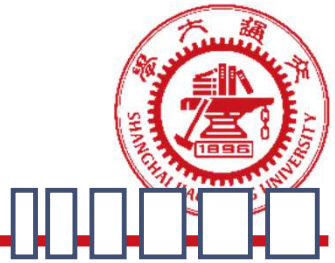
- 噪声对应于高频部分，所以去噪可以采用低通滤波。
- 频域低通滤波的核心关键为

---------设计合适的低通滤波器H(u,v)

$$G(u,v)=H(u,v)F(u,v)$$



6.3 频域平滑滤波



6.3.1 理想低通滤波

6.3.2 巴特沃斯低通滤波

6.3.3 指数低通滤波

6.3.4 梯形低通滤波

6.3.1理想低通滤波

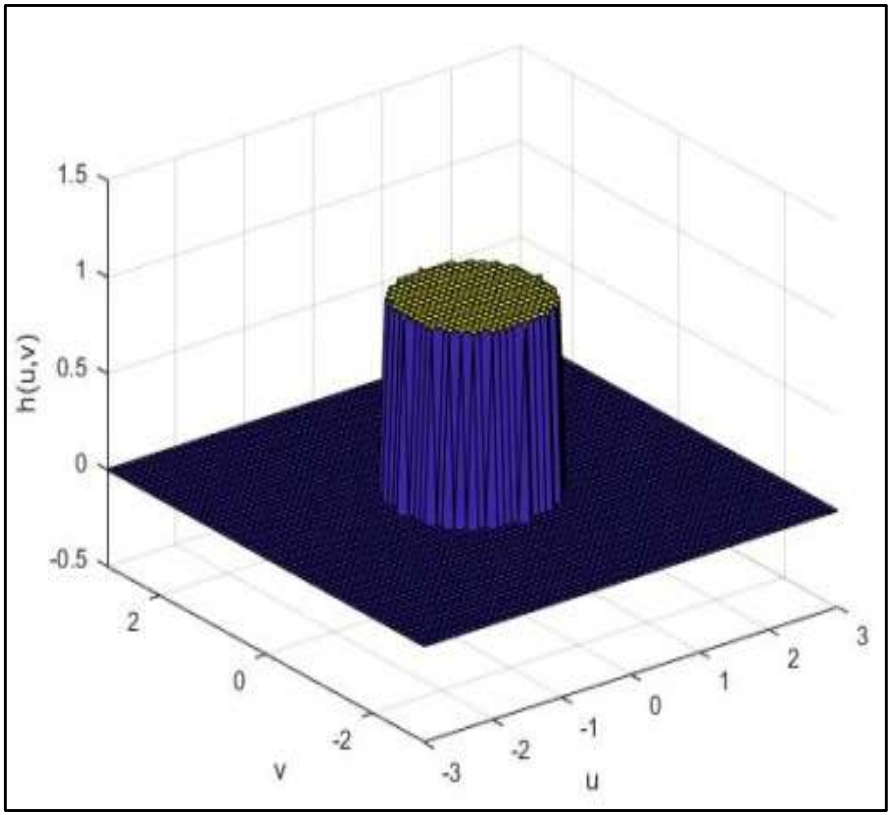
（1）原理

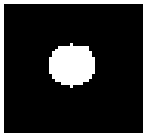
$$H(u,v)=\begin{cases}1 & D(u,v)\le D_0\\ 0 & D(u,v)>D_0\end{cases}$$

- D_0为截止频率
- $D(u,v)=(u^2+v^2)^{1/2}$为频率平面原点到点(u,v)距离
- 特点
  - 物理上不可实现
  - 有振铃现象
  - 滤除高频成分使图像严重模糊

6.3.1理想低通滤波

理想低通滤波器转移函数的透视图

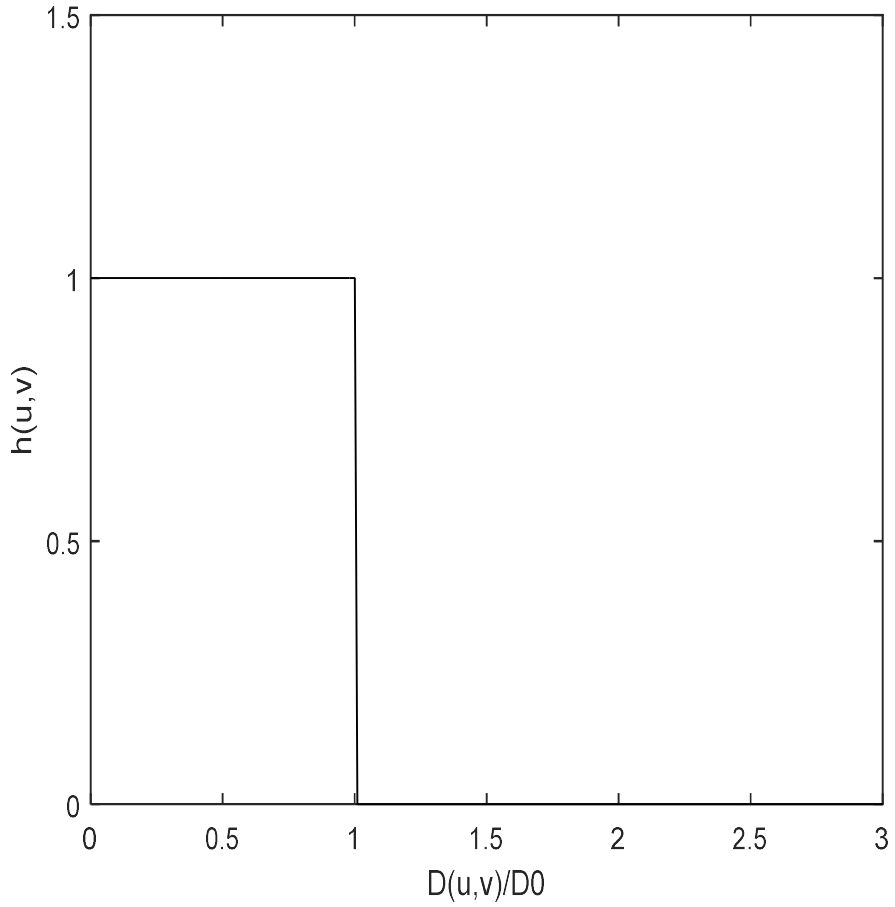




ILPF在截止频率直上直下，在物理上不可实现。

6.3.1理想低通滤波

理想低通滤波器转移函数的剖面图



“振铃”——

- H(u,v)在D_0处由1突变到0，其对应的冲激响应h(x,y)在空域中表现为同心环形式。
- 同心环半径与D_0成反比。
- D_0越小，同心环半径越大，模糊越厉害。

6.3.1理想低通滤波

- 振铃现象

（2）例程---截断频率不同的理想低通滤波器

- 程序

```
Image=imread('lena.bmp');
FImage=fftshift(fft2(double(Image)));
[N M]=size(FImage);
g=zeros(N,M);
r1=floor(M/2);  r2=floor(N/2);
d0=[5 11 45 68];
for i=1:4
    for x=1:M
        for y=1:N
```

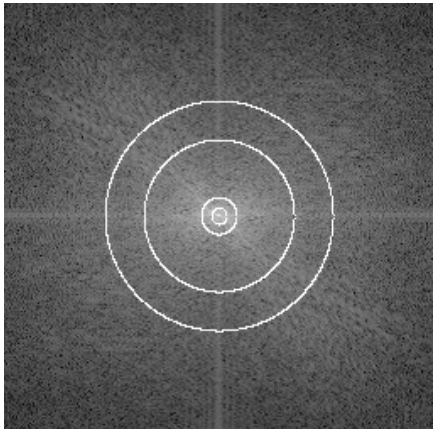
6.3.1理想低通滤波


```
            d=sqrt((x-r1)^2+(y-r2)^2);
             if d<=d0(i)
                h=1;
             else
                h=0;
             end
             g(y,x)=h*FImage(y,x);
          end
       end
       g= real(ifft2(ifftshift(g)));
       figure,imshow(uint8(g)),
              title(['理想低通滤波D0=',num2str(d0(i))]);
   end
```

6.3.1理想低通滤波

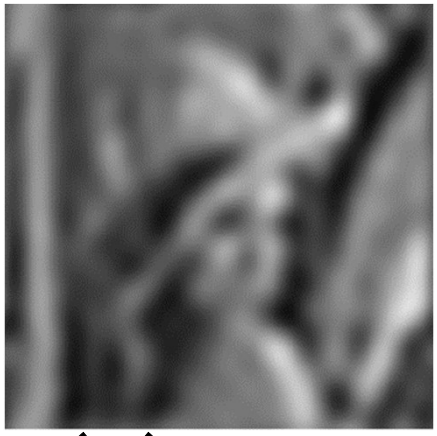
- 效果

（a）*lena*图像



（b）傅里叶频谱

（c）D_0=5,能量为90%



（d）D_0=11,
能量为95%

（e）D_0=45,
能量为99%

（f）D_0=68,
能量为99.5%

6.3.2巴特沃斯低通滤波

（1）原理

又称为最大平坦滤波器。一个阶为n，截断频率为D_0的巴特沃思低通滤波器转移函数：

$$H(u,v)=\frac{1}{1+\left[D(u,v)/D_0\right]^{2n}}$$

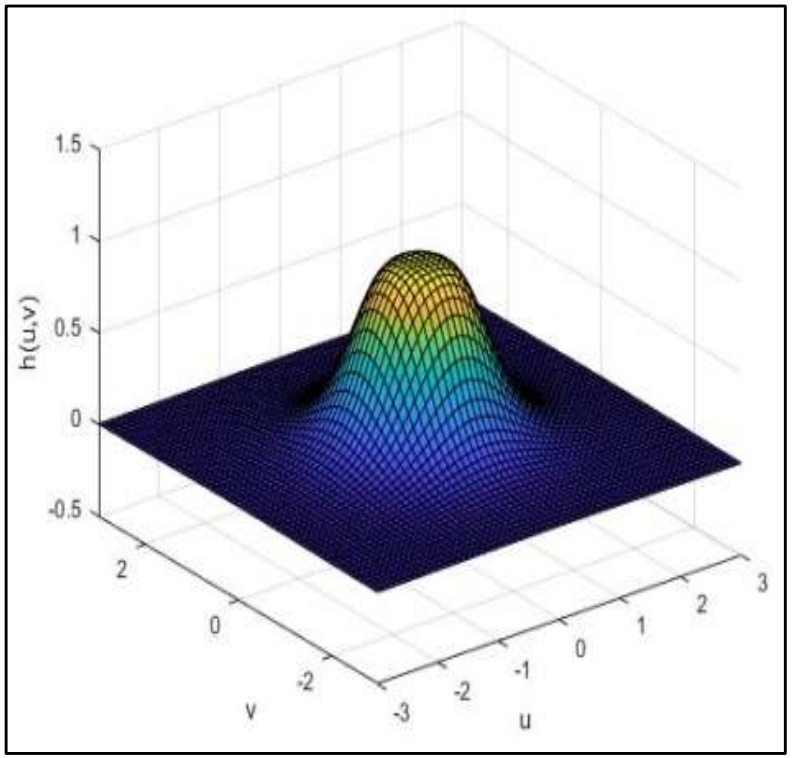
或

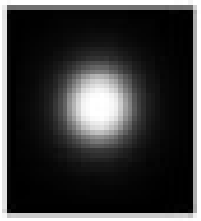
$$H(u,v)=\frac{1}{1+(\sqrt{2}-1)[D(u,v)/D_0]^{2n}}$$

当$D(u,v)=D_0$时，$H(u,v)$降为最大值的$\frac{1}{2}$或$\frac{1}{\sqrt{2}}$

6.3.2巴特沃斯低通滤波

BLPF转移函数的透视图

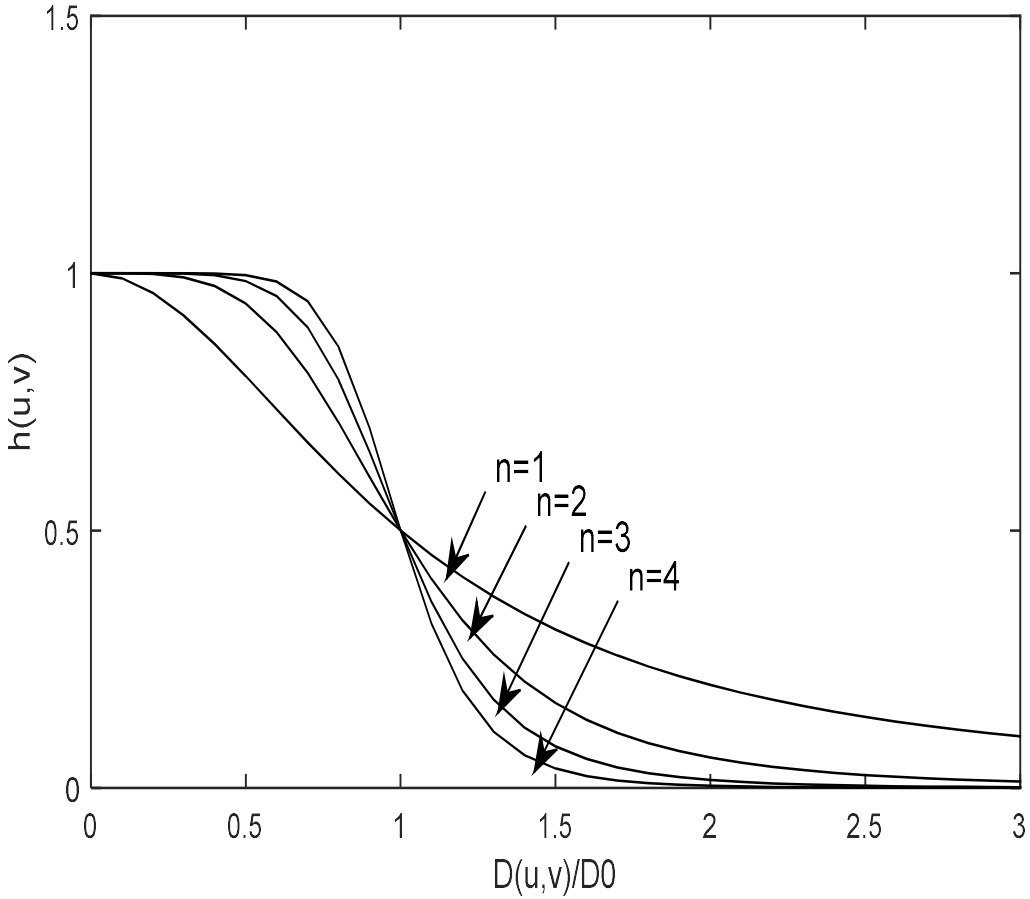




- 特性
  - 是连续衰减
  - 图像边缘的模糊程度大大减小
  - 对噪声的平滑效果不如ILPF
  - 振铃效应减缓

6.3.2巴特沃斯低通滤波

（不同阶数n）BLPF转移函数的剖面图



随着阶数*n*的增加，图像振铃效应越来越明显。

（2）例程---阶数不同的巴特沃斯低通滤波器

- 程序

```
Image=imread('lena.bmp');
Image=imnoise(Image,'gaussian');
FImage=fftshift(fft2(double(Image)));
[N M]=size(FImage);
g=zeros(N,M);
r1=floor(M/2);  r2=floor(N/2);
d0=30;
n=[1 2 3 4];
for i=1:4
```

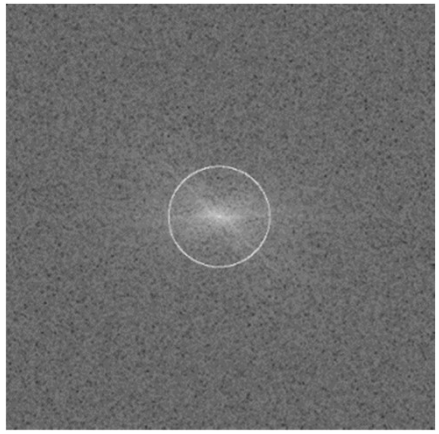
6.3.2巴特沃斯低通滤波


```
    for x=1:M
        for y=1:N
            d=sqrt((x-r1)^2+(y-r2)^2);
            h=1/(1+(d/d0)^(2*n(i)));
            g(y,x)=h*FImage(y,x);
        end
    end
    g=ifftshift(g);
    g=real(ifft2(g));
figure,imshow(uint8(g)),
        title(['Butterworth低通滤波n=',num2str(n(i))]);
end
```

6.3.2巴特沃斯低通滤波

- 效果

（a）高斯噪声



（b）傅里叶频谱

（c） n=1，D_0=30

（d）n=2，D_0=30

（e）n=3，D_0=30

（f）n=4，D_0=30

6.3.2巴特沃斯低通滤波

D_0=20

D_0=35

- 噪声点被有效地去除，但图像也变得模糊。
- 随着阶数*n*的增加，图像振铃效应越来越明显。

6.3.3指数低通滤波

（1）原理

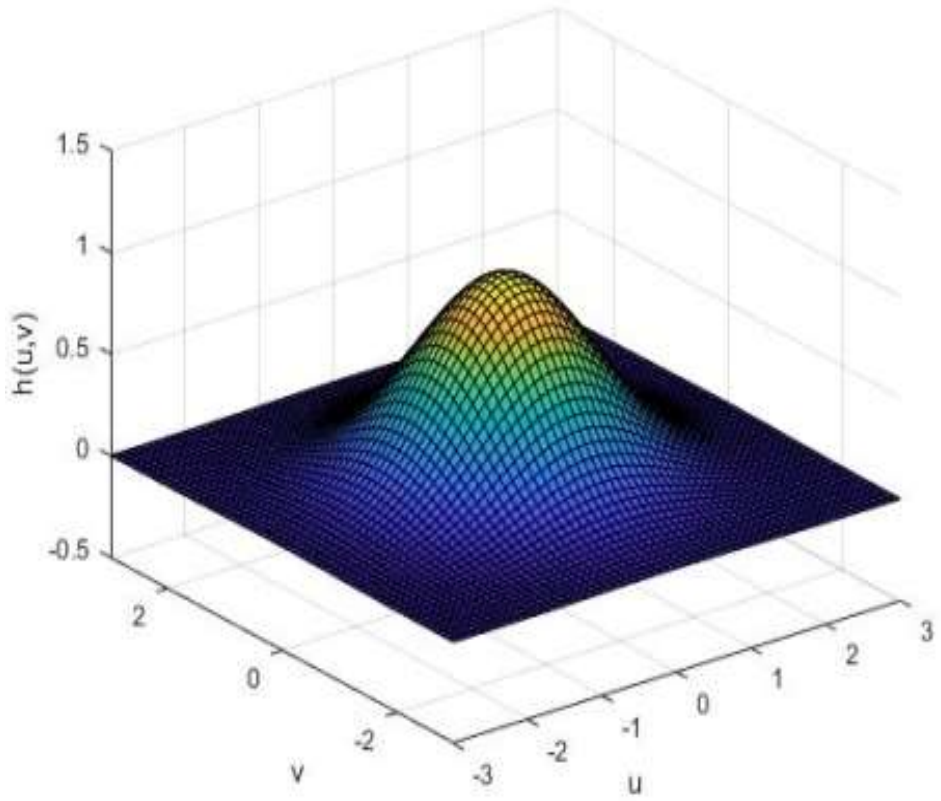
截断频率为D_0的指数低通滤波器转移函数：

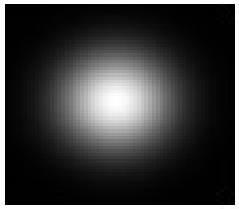
$$H(u,v)=e^{-\frac{D^2(u,v)}{2D_0^2}}$$

当$D(u,v)=D_0$时，$H(u,v)$降为最大值的0.607处

6.3.3指数低通滤波

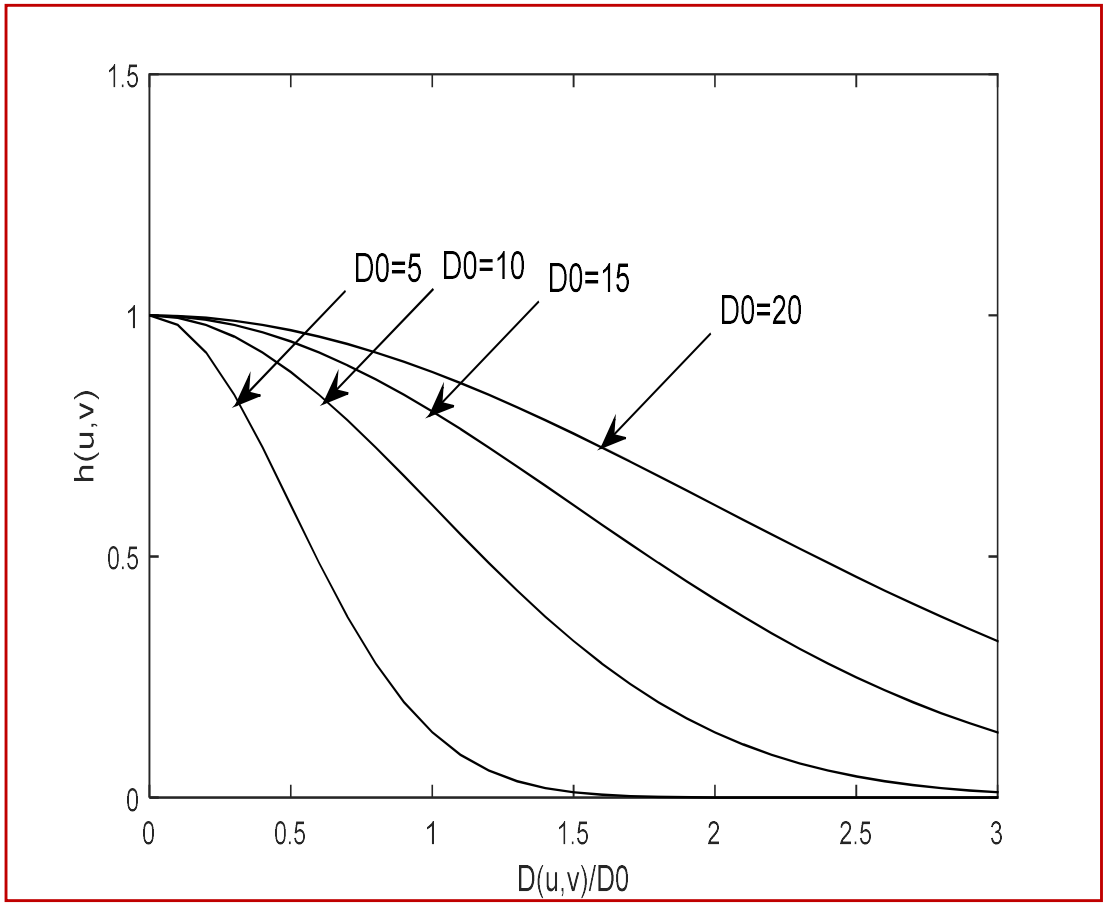
指数低通滤波器转移函数的透视图





6.3.3指数低通滤波

不同截止频率的ELPF转移函数的剖面图



随着截止频率D_0的增加，图像振铃效应越来越不明显。

6.3.3指数低通滤波

（2）例程---截止频率不同的指数低通滤波器

- 程序

```
Image=imread('lena.bmp');
Image=imnoise(Image,'gaussian');
FImage=fftshift(fft2(double(Image)));
[N M]=size(FImage);
g=zeros(N,M);
r1=floor(M/2);  r2=floor(N/2);
d0=[20 40];
n=2;
for i=1:2
```

6.3.3指数低通滤波


```
    for x=1:M
        for y=1:N
            d=sqrt((x-r1)^2+(y-r2)^2);
            h=exp(-0.5*(d/d0(i))^n);
            g(y,x)=h*FImage(y,x);
        end
    end
    g=ifftshift(g);
    g=real(ifft2(g));
figure,imshow(uint8(g)),
        title(['指数低通滤波D0=',num2str(d0(i))]);
end
```

6.3.3指数低通滤波

- 效果



（a）高斯噪声　（b）n=2，D_0=20　（c）n=2，D_0=40

- 噪声点被有效地去除；
- 随着截止频率D_0的增加，图像模糊现象减弱；
- 与巴特沃斯低通滤波器相比较，指数低通滤波器没有振铃现象。

6.3.4梯形低通滤波

（1）原理

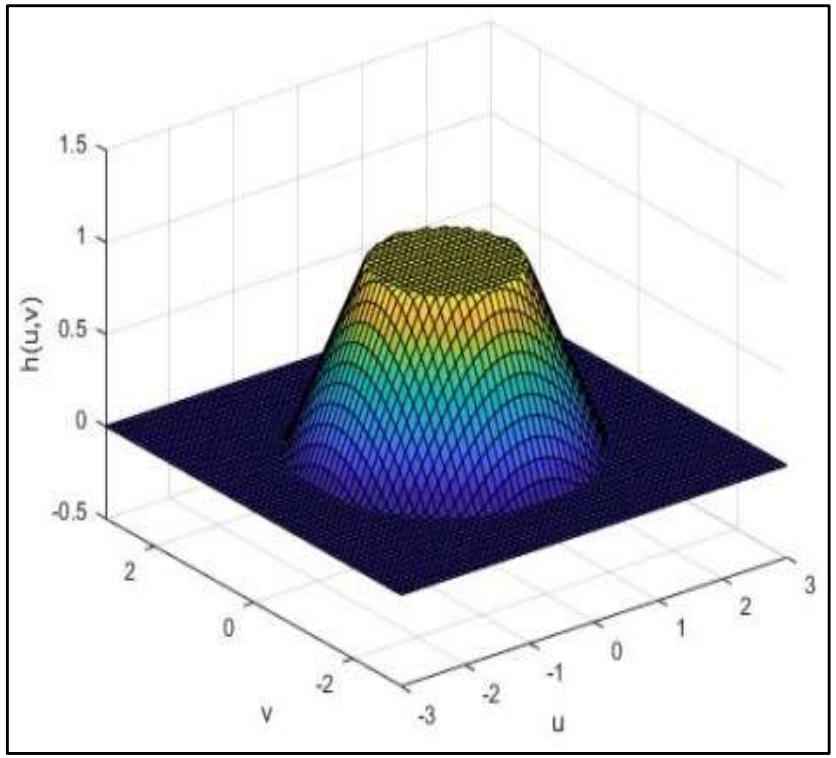
梯形低通滤波器（TLPF）的传递函数：

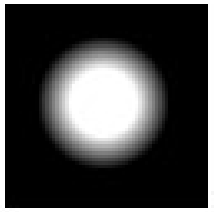
$$H(u,v)=\begin{cases}1, & D(u,v)\le D_0\\ \dfrac{D(u,v)-D_1}{D_0-D_1}, & D_0<D(u,v)\le D_1\\ 0 & D(u,v)>D_1\end{cases}$$

D_0为截止频率，D_0、D_1需满足：$D_0<D_1$

6.3.4梯形低通滤波

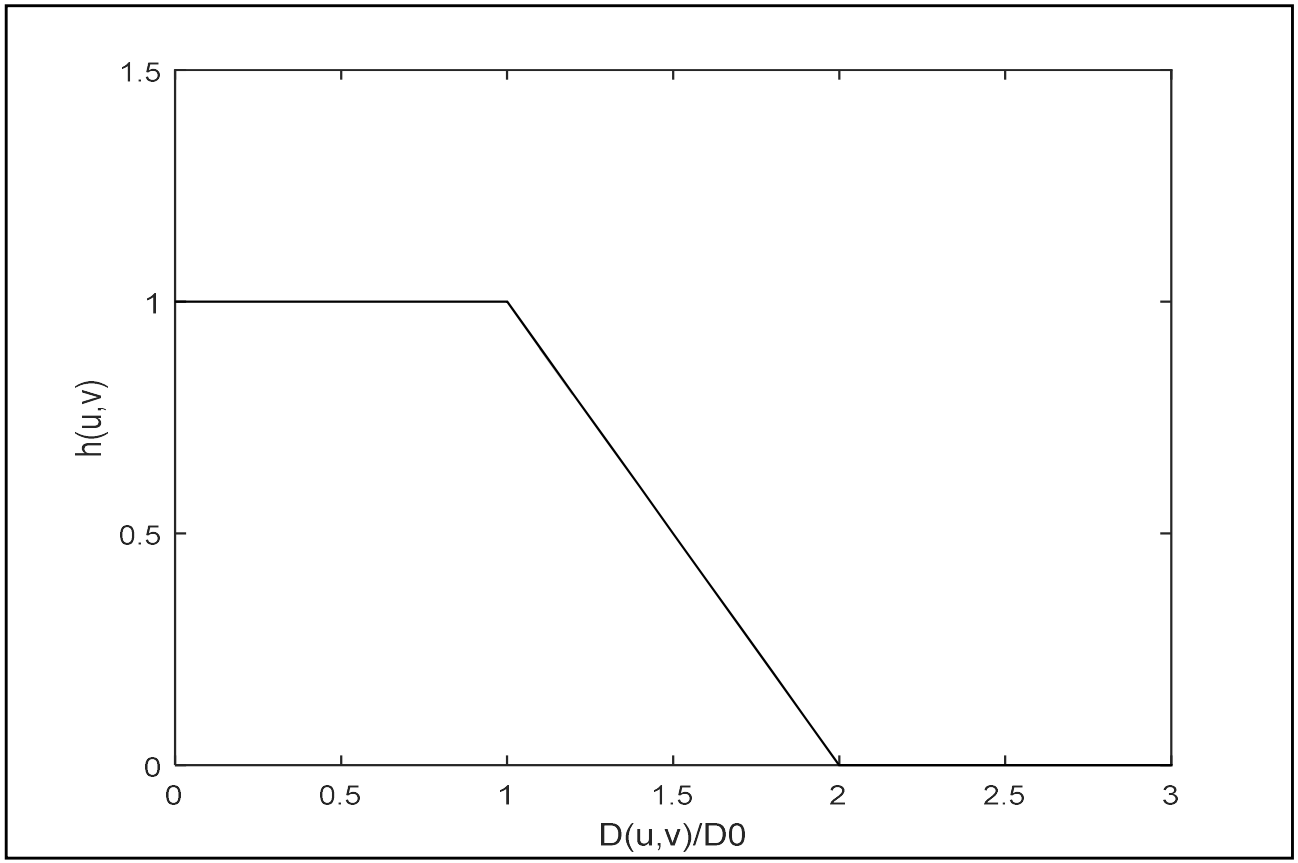
梯形低通滤波器转移函数的透视图





6.3.4梯形低通滤波

梯形低通滤波器转移函数的剖面图



6.3.4梯形低通滤波

（2）例程---截止频率不同的梯度低通滤波器

- 程序

```
Image=imread('lena.bmp');
Image=imnoise(Image,'gaussian');
FImage=fftshift(fft2(double(Image)));
[N M]=size(FImage);
g=zeros(N,M);
r1=floor(M/2);  r2=floor(N/2);
d0=[5 30];  d1=[45 70];
for i=1:2
    for x=1:M
        for y=1:N
```

6.3.4梯形低通滤波


```
        d=sqrt((x-r1)^2+(y-r2)^2);
        if d>d1
            h=0;
        else
            if d>d0
                h=(d-d1)/(d0-d1);
            else
                h=1;
            end
        end
        g(y,x)=h*FImage(y,x);
    end
end
```

6.3.4梯形低通滤波


```
g=ifftshift(g);
g=real(ifft2(g));
figure,imshow(uint8(g)),title(['梯度低通滤波D0=',
                num2str(d0(i)),',D1=',num2str(d1(i))]);
end
```

6.3.4梯形低通滤波

- 效果

（a）高斯噪声

（b）D_0=5，D_1=45

（c）D_0=30，D_1=70

- 噪声点被有效地去除；
- 与指数低通滤波器相比较，滤波输出图像有一定的模糊和振铃效应。

6.4 其他图像平滑方法

6.4.1 基于模糊技术的平滑滤波

6.4.2 基于偏微分方程的平滑滤波

6.4.1基于模糊技术的平滑滤波

其他图像平滑方法

- 在图像处理中，可以将一幅图像看成一个模糊集。
- 当图像被噪声高度污染时，其模糊不确定性增加，可应用模糊滤波来处理图像。
- 基于模糊数学思想，利用模糊隶属度函数的概念，通过对均值滤波器的权值加以优化，提高平滑高斯噪声的能力。

6.4.1基于模糊技术的平滑滤波

- 模糊加权均值滤波的具体算法步骤：
  - 计算以点(x,y)为中心的邻域S内灰度变化

$$\beta(m,n)=\frac{1}{N}\sum_{(m,n)\in S}d(m,n)$$

$$d(m,n)=\left(f(x,y)-f(m,n)\right)^2,\quad \left((m,n)\in S\text{且}(m,n)\neq(x,y)\right)$$

  - 计算每一邻域点对中心点的模糊隶属度

$$\mu(m,n)=e^{\left(\frac{-d(m,n)}{\beta(m,n)}\right)}$$

其中，$\mu(m,n)$为模糊隶属度，$0\leq\mu(m,n)\leq 1$

6.4.1基于模糊技术的平滑滤波

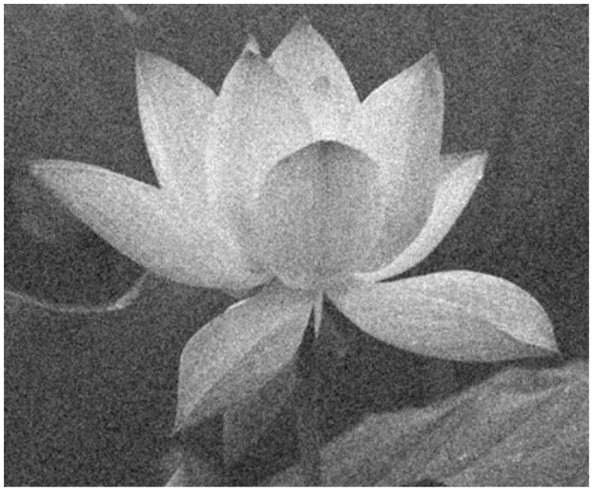
- 计算当前窗口模糊加权均值滤波输出

$$g(x,y)=\frac{\sum_{N}\frac{\mu(m,n)}{\beta(m,n)}\cdot f(m,n)}{\sum_{N}\frac{\mu(m,n)}{\beta(m,n)}}$$

- 用加权平均值 $g(x,y)$ 来代替滤波窗口内中心像素点的灰度值，实现对高斯噪声点的滤波处理。

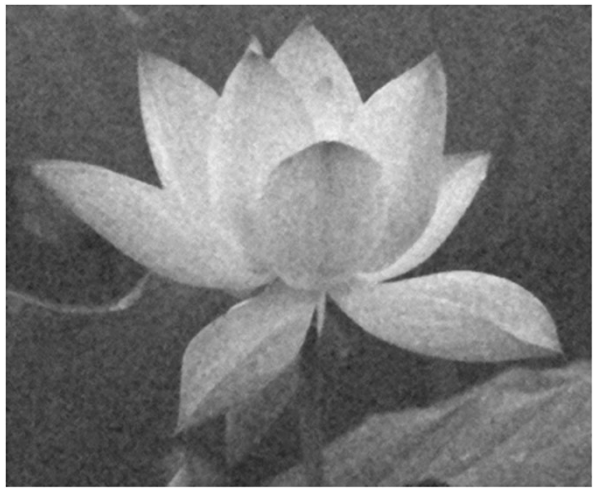
6.4.1基于模糊技术的平滑滤波

- 效果



(a)高斯噪声图像

(b)5×5均值滤波



(c)5×5模糊加权均值滤波

6.4.2基于偏微分方程的平滑滤波

其他图像平滑方法

- 假设灰度图像μ_0: $R_2 \to R$，引入时间参数t，表示为$\mu(x,y;t)$，且有: $\mu(x,y;0)=\mu_0(x,y)$。
- 给出一种规定算法对应的偏微分算子F: $R \to R$，根据F定义不同有：线性扩散、非线性扩散、各向异性扩散等。
- 对图像的处理以偏微分方程表示可写为

$$\frac{\partial u}{\partial t} = F\left[u(x,y;t)\right]$$

偏微分方程的解$\mu(x,y;t)$给出迭代t次时的图像，最终在得到满意的图像时停止迭代。

6.4.2基于偏微分方程的平滑滤波

（1）基于偏微分方程的平滑滤波分类

- 基于多尺度分析的PDE模型——

  典型的“P-M模型”

- 基于变分的PDE模型——

典型的“基于全变分法的PDE图像去噪模型”

- 基于几何制约的PDE模型——典型的“曲率流”，是“纯粹的”各向异性扩散模型
- 小波变换与非线性PDE相结合的模型。

（2）各向同性热扩散模型

对图像的各向同性扩散以经典的热传导方程为

$$\begin{cases} \dfrac{\partial u(x,y;t)}{\partial t} = \Delta u(x,y;t) \\ u_0 = u(x,y;0) \end{cases}$$

求解

$$\begin{cases} u(x,y;0) = u_0 & t = 0 \\ u(x,y;t) = G_t(x,y) * u(x,y;0) & t \geq 0 \end{cases}$$

其中，$G_t(x,y)$为高斯核函数

6.4.2基于偏微分方程的平滑滤波

（2）各向同性热扩散模型

$$G_t(x,y)=\frac{1}{4\pi t}e^{-\frac{x^2+y^2}{4t}}$$

- t 代表一个尺度参数，对应的是迭代时间。
- 选择不同的迭代时间，即得到不同尺度下的平滑图像。

- 各向同性扩散由于没有考虑图像空间位置，在整个图像支撑集采用同样的平滑策略，因此在平滑去噪过程中，会带来边界模糊。

6.4.2基于偏微分方程的平滑滤波

（3）各向异性P-M非线性扩散模型

- 是由Perona 和 Malik 在1990年提出；
- 是通过表征图像边缘的梯度值自动调节扩散系数来实现控制不同区域的平滑程度。
- **P-M的非线性扩散方程为**

$$\begin{cases} \dfrac{\partial u(x,y;t)}{\partial t} = div\left[g\left(\left\| \nabla u(x,y;t) \right\| \right) \nabla u(x,y;t) \right] \\ u(x,y;0) = u_0 \end{cases}$$

其中，div为散度算子，∇为梯度算子，$\|\|$ 表示幅度

6.4.2基于偏微分方程的平滑滤波

$g(\|\nabla u(x,y;t)\|)$为扩散系数方程，是一个非负的以梯度幅度值为自变量的减函数

$$g\left(\left\|\nabla u(x,y;t)\right\|\right)=\frac{1}{1+\left(\frac{\left\|\nabla u(x,y;t)\right\|}{k}\right)^2}$$

或

$$g\left(\left\|\nabla u(x,y;t)\right\|\right)=e^{-\left(\frac{\left\|\nabla u(x,y;t)\right\|}{k}\right)^2}$$

其中，k为梯度阈值。

6.4.2基于偏微分方程的平滑滤波

- **P-M模型离散表达式**

$$u_{i,j}^{t+1}=u_{i,j}^{t}+\lambda\left[g_{S_{i,j}}^{t}\cdot\nabla_S u_{i,j}^{t}+g_{E_{i,j}}^{t}\cdot\nabla_E u_{i,j}^{t}+g_{N_{i,j}}^{t}\cdot\nabla_N u_{i,j}^{t}+g_{W_{i,j}}^{t}\cdot\nabla_W u_{i,j}^{t}\right]$$

$$\begin{cases} u_{i,j}^{t}=u(i,j,t); \quad u_{i,j}^{t+1}=u(i,j,t+1); \\ g_{S_{i,j}}^{t}=g\left(\left\|\nabla_S u_{i,j}^{t}\right\|\right); \quad g_{E_{i,j}}^{t}=g\left(\left\|\nabla_E u_{i,j}^{t}\right\|\right); \quad g_{N_{i,j}}^{t}=g\left(\left\|\nabla_N u_{i,j}^{t}\right\|\right); \quad g_{W_{i,j}}^{t}=g\left(\left\|\nabla_W u_{i,j}^{t}\right\|\right) \\ \nabla_S u_{i,j}^{t}=u_{i,j-1}^{t}-u_{i,j}^{t}; \quad \nabla_E u_{i,j}^{t}=u_{i-1,j}^{t}-u_{i,j}^{t}; \quad \nabla_N u_{i,j}^{t}=u_{i,j+1}^{t}-u_{i,j}^{t}; \quad \nabla_W u_{i,j}^{t}=u_{i+1,j}^{t}-u_{i,j}^{t}; \end{cases}$$

- λ为控制扩散总体强度的常数， $\lambda>0$。
- S, E, N, W 表示点(i, j) 的4-邻域

6.4.2基于偏微分方程的平滑滤波

- 效果

（a）高斯噪声图像

（b）迭代5次

（c）迭代15次

即使迭代时间增加，
算法在去噪同时且能
较好地保护图像边缘