第九讲：图像复原

任课老师：寇旗旗

TEL:18852146321 QQ:137156449

中国矿业大学计算机科学与技术学院

问题的提出：

- 在图像生成、记录、传输过程中，由于成像系统、设备或外在的干扰，会导致图像质量下降，称为图像退化，如大气扰动效应、光学系统的像差、物体运动造成的模糊、几何失真等。
- 对退化图像进行处理，使之恢复原貌的技术称之为图像复原（Image Restoration）。
- 图像复原的关键在于确定退化的相关知识，将退化过程模型化，采用相反的过程尽可能恢复原图，或者说使复原后的图像尽可能接近原图。

前言

图像复原与图像增强的研究内容有一定的交叉性：

1. 图像增强是一种改进图像视觉效果的技术。不考虑图像是如何退化的，而试图采用各种技术来增强图像的视觉效果。因此，图像增强可以不顾增强后的图像是否失真，只要看得舒服就行。
2. 图像复原是一种对退化（或品质下降）了的图像去除退化因素，进而恢复或重建被退化了的图像的技术。力求保持图像本来面目，以保真原则为前提，找出图像降质的原因，描述其物理过程，提出数学模型。根据该模型重建或恢复被退化的图像。

前言

图像增强与复原

增强

复原

主要内容

9.1 图像退化模型

9.2 图像退化函数的估计

9.3 图像复原的代数方法

9.4 典型图像复原方法

9.5 盲去卷积复原

9.6 几何失真校正

9.1 图像退化模型

9.1.1 连续退化模型

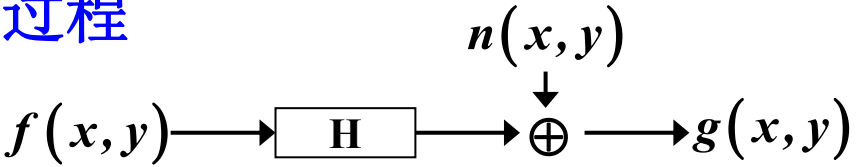
9.1.2 离散退化模型

9.1.3 图像复原

9.1.4 图像去噪

9.1.1 连续退化模型

（1）退化过程



抽象为一个退化系统H以及加性噪声的影响

$$g(x,y)=H\left[f(x,y)\right]+n(x,y)$$

用线性、空间不变系统模型来模拟实际中的非线性和空间变化模型

$$H\left[f(x,y)\right]=f(x,y)*h(x,y)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}f(\alpha,\beta)h(x-\alpha,y-\beta)d\alpha d\beta$$

（2）退化模型

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

$h(x,y)$称为点扩散函数（PSF），其傅里叶变换$H(u,v)$也称为光学传递函数（OTF）

9.1.2 离散退化模型

$f(\alpha,\beta)h(x-\alpha,y-\beta)$ 进行均匀取样得到离散退化模型

- 二维离散卷积退化模型

$$g_e(x,y)=\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}f_e(m,n)h_e(x-m,y-n),x=0\sim M-1;y=0\sim N-1$$

9.1.3 图像复原

（1）含义

图像复原是指在给定退化图像 $g(x,y)$，了解退化的点扩散函数 $h(x,y)$ 和噪声项 $n(x,y)$ 的情况下，估计出原始图像 $f(x,y)$

（2）步骤

- 确定图像的噪声类型和退化函数

  退化函数一般是不知道的，需先估计退化函数

- 采用合适的图像复原方法复原图像

噪声滤除后，采用与退化相反的过程，使复原后的图像尽可能接近原图，一般要确定一个合适的准则函数，最优情况对应最好的复原图。这一步的关键技术在于确定准则函数和求最优。

可采用盲复原方法：直接从退化图像估计原图像

9.1.4 图像噪声

数字图像中的噪声源来自于图像获取（AD转换，即将连续转为数字）以及传输过程：

- 图像传感器会受到环境的干扰
- 图像在传输过程中会受到的干扰

9.1.4 图像噪声

- 噪声与图像的相关性
  - 相关——乘性噪声
  - 不相关——加性噪声

- **白噪声**
  - 图像平面上不同点的噪声是不相关的，其谱密度为常数。
  - 一般假设图像上的噪声是白噪声。
  - 实用上，只要噪声带宽远大于图像带宽，就可把它当作白噪声。

9.1.4 图像噪声

一些重要噪声的概率密度函数

- 高斯噪声
- 瑞利噪声
- 均匀分布噪声
- 脉冲噪声（椒盐噪声）
- 伽马（爱尔兰）噪声
- 指数分布噪声

9.1.4 图像噪声

常见的噪声及其概率密度函数

（1）高斯噪声

高斯噪声是一种源于**电子电路**噪声和由**低照明度或高温**带来的**传感器噪声**。高斯噪声也称为**正态噪声**，其概率密度函数为：

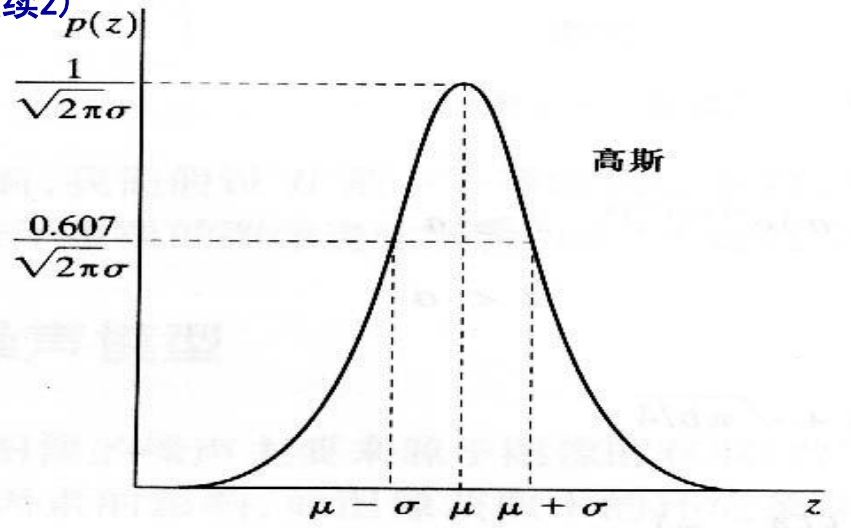
$$p(z)=\frac{1}{\sqrt{2\pi}\sigma}e^{-(z-\mu)^2/2\sigma^2}$$

概率密度函数

其中：高斯随机变量z表示灰度值；μ表示z的平均值或者期望值；σ表示z的**标准差**，而标准差的平方σ^2称为z的方差。

9.1.4 图像噪声

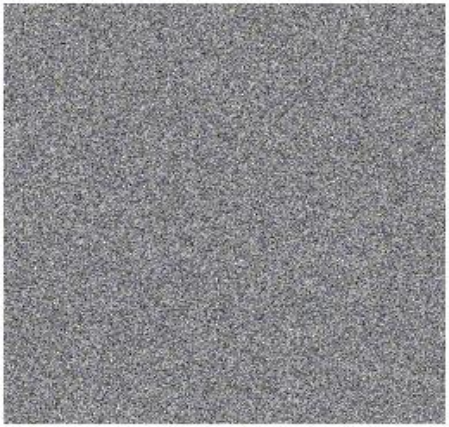
（1）高斯噪声 (续2)



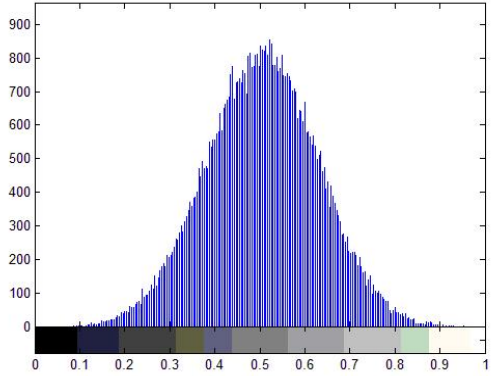
高斯噪声概率密度函数随频率呈高斯分布。所谓白噪声，是指图像面上不同点的噪声是不相关的，其功率谱为常量（均匀分布），也即其强度不随频率的增加而衰减。

9.1.4 图像噪声

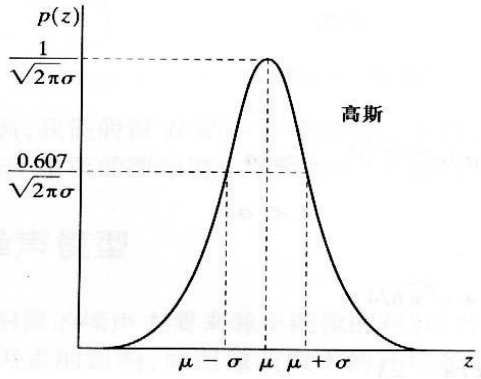
（1）高斯噪声图像（直方图为高斯分布的图像）



高斯噪声

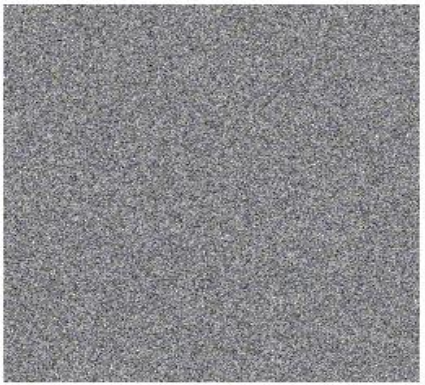


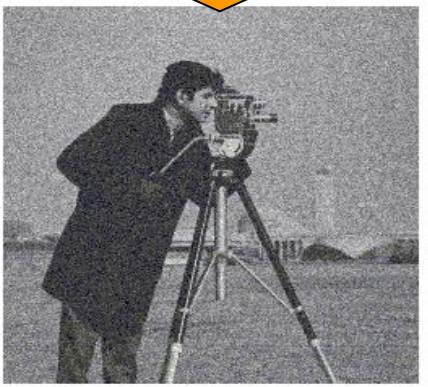
直方图



9.1.4 图像噪声

（1）高斯噪声污染的图像





9.1.4 图像噪声

（2）瑞利噪声

瑞利噪声的概率密度函数为：

$$p(z)=\begin{cases}\dfrac{2}{b}(z-a)e^{-(z-a)^2/b} & z\geq a\\ 0 & z<a\end{cases}$$

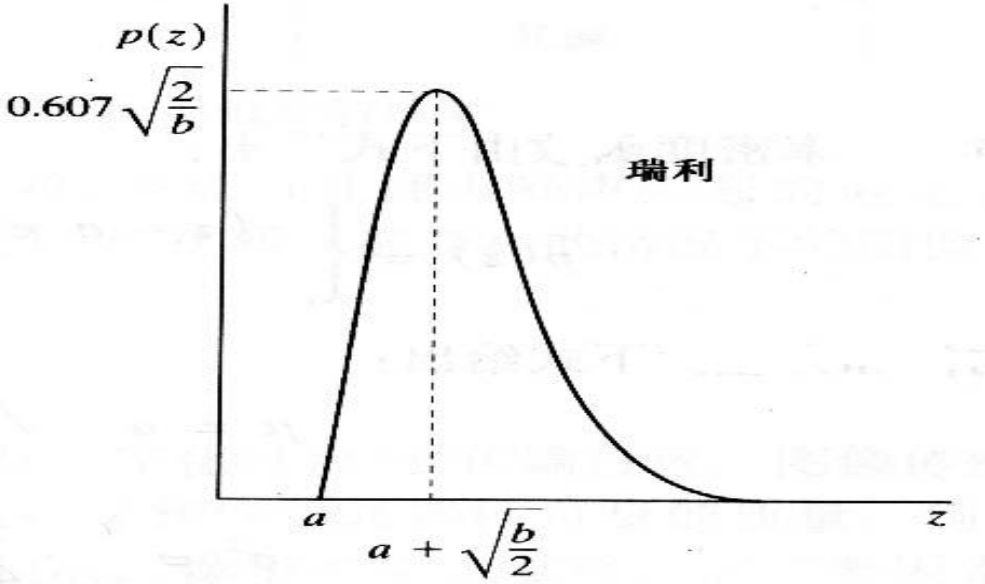
概率密度的均值和方差分别为：

$$\mu=a+\sqrt{\pi\cdot b/4}$$ 均值

$$\sigma^2=\frac{b(4-\pi)}{4}$$ 方差

9.1.4 图像噪声

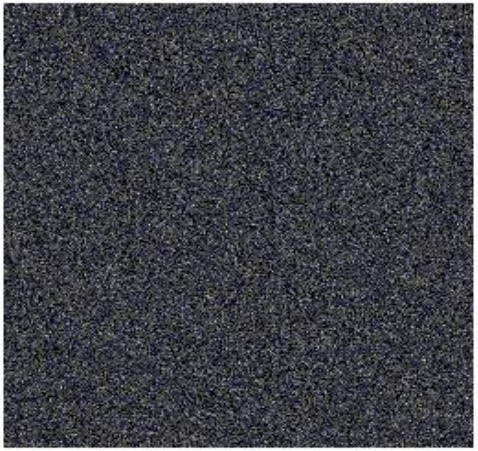
（2）瑞利噪声 （续2）

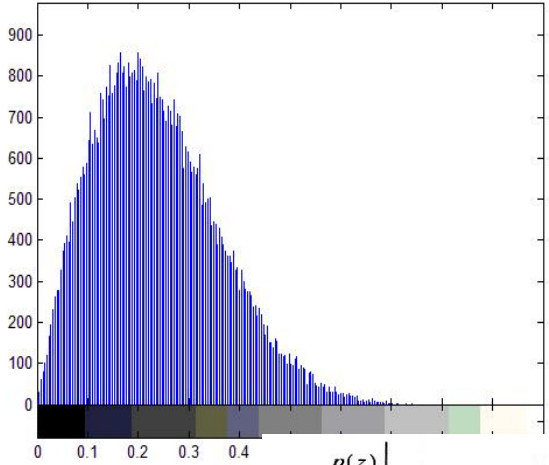


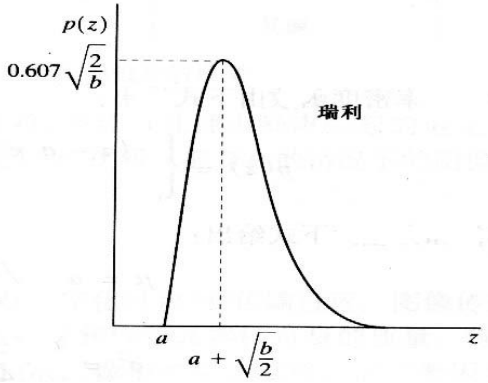
概率密度函数

9.1.4 图像噪声

（2）瑞利噪声图像（直方图为瑞利分布的图像）

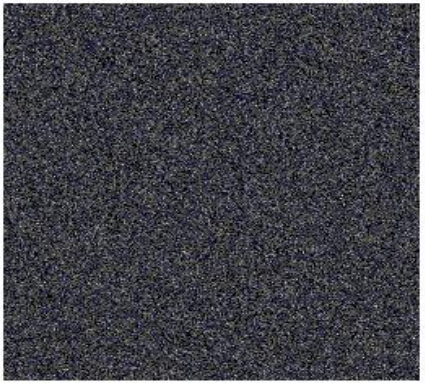


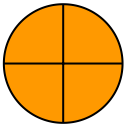




9.1.4 图像噪声

（2）瑞利噪声污染的图像





9.1.4 图像噪声

（3）均匀分布噪声

均匀分布噪声的概率密度函数为：

$$p(z)=\begin{cases}\dfrac{1}{b-a} & a\le z\le b\\ 0 & \text{其它}\end{cases}$$

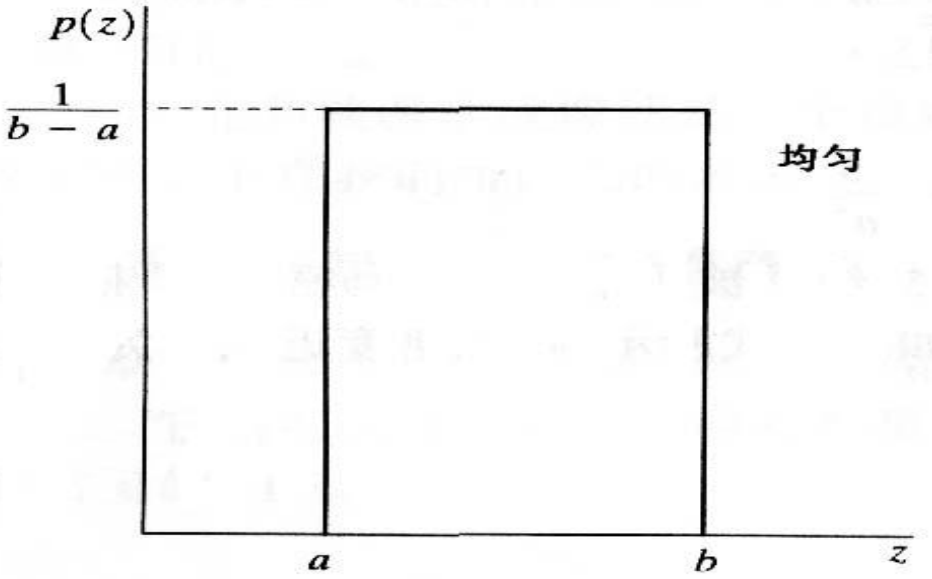
概率密度的期望值和方差分别为：

$$\mu=\frac{a+b}{2}$$ 数学期望

$$\sigma^2=\frac{(b-a)^2}{12}$$ 方差

9.1.4 图像噪声

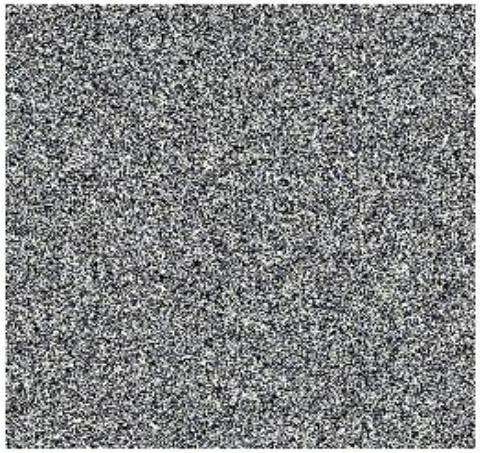
（3）均匀分布噪声 （续2）

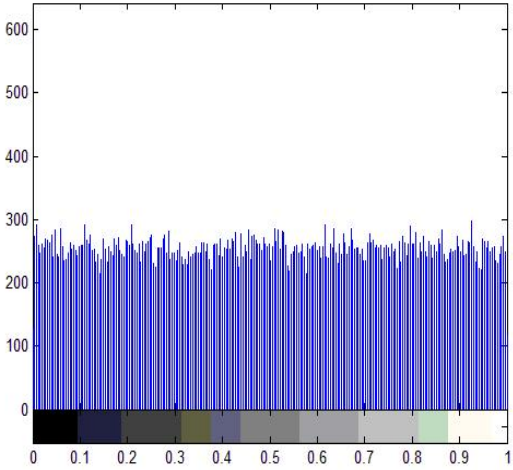


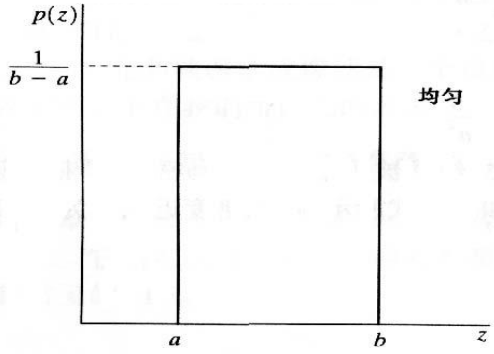
概率密度函数

9.1.4 图像噪声

（3）均匀噪声的图像（直方图为均匀分布的图像）

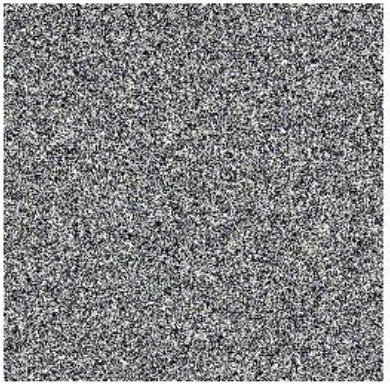


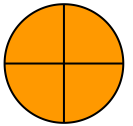




9.1.4 图像噪声

（3）均匀噪声污染的图像





9.1.4 图像噪声

（4）脉冲噪声（椒盐噪声）

（双极）脉冲噪声的概率密度为：

$$p(z)=\begin{cases}P_a & z=a\\ P_b & z=b\\ 0 & \text{其它}\end{cases}$$

表示的脉冲噪声在P_a或P_b均不可能为零，且在脉冲可能是正值、也可能是负值的情况下，称为双极脉冲噪声。

通常，负脉冲以黑点（胡椒点）出现，正脉冲以白点（盐点）出现

9.1.4 图像噪声

（4）脉冲噪声（椒盐噪声）(续2)

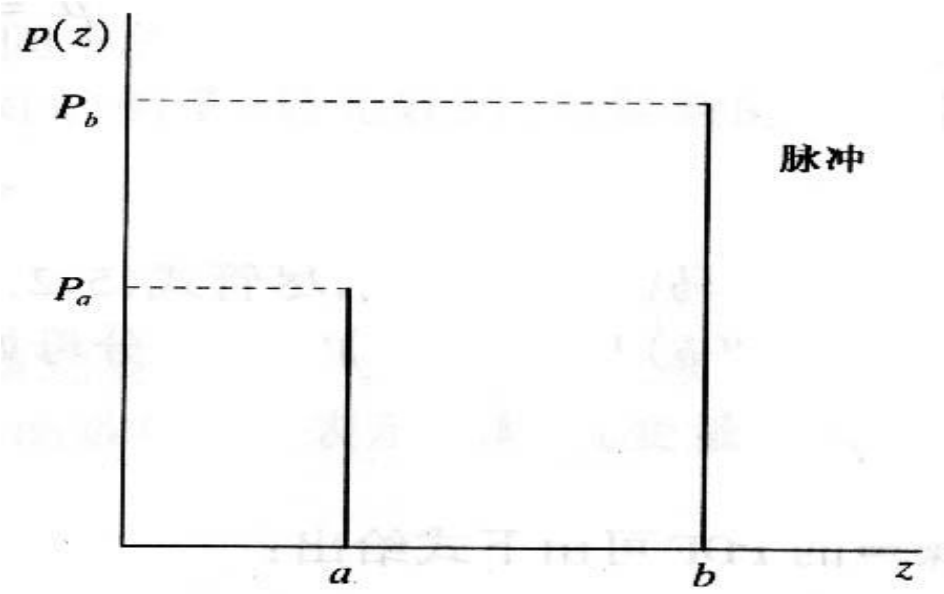
说明：

（1）如果：所表示的脉冲噪声在P_a或P_b其中之一为零，则脉冲噪声称为**单极脉冲噪声**；

（2）通常情况下脉冲噪声总是数字化为允许的最大值或最小值，所以**负脉冲以黑点**(胡椒点)出现在图像中，**正脉冲以白点**（盐点）出现在图像中。

9.1.4 图像噪声

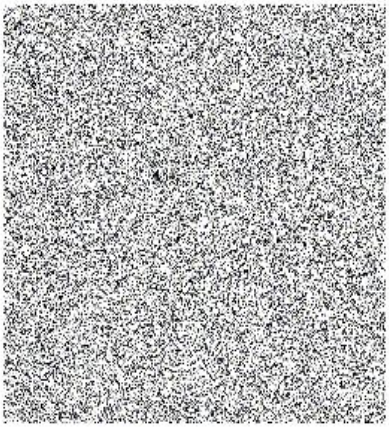
（4）脉冲噪声（椒盐噪声）(续3)

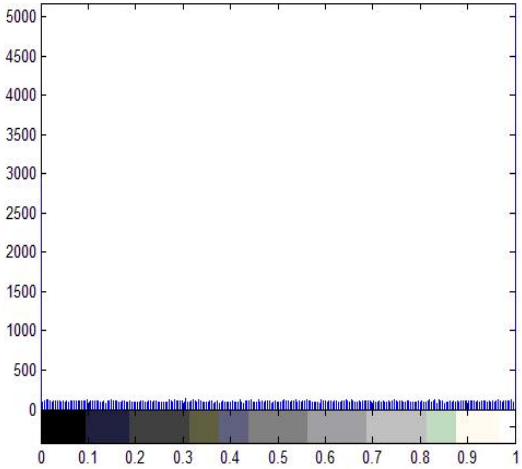


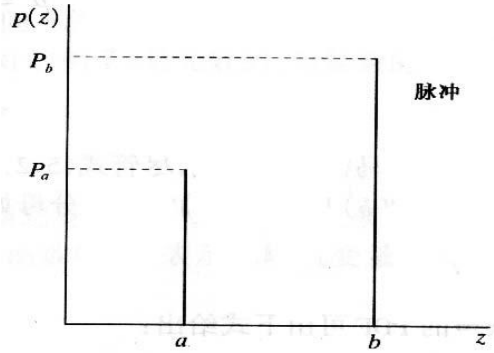
概率密度函数

9.1.4 图像噪声

（4）脉冲噪声的图像（直方图为脉冲的图像）

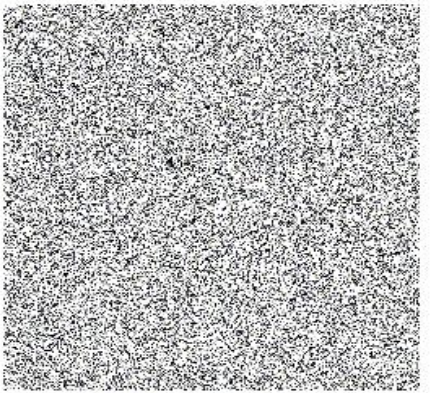


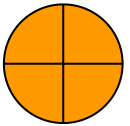




9.1.4 图像噪声

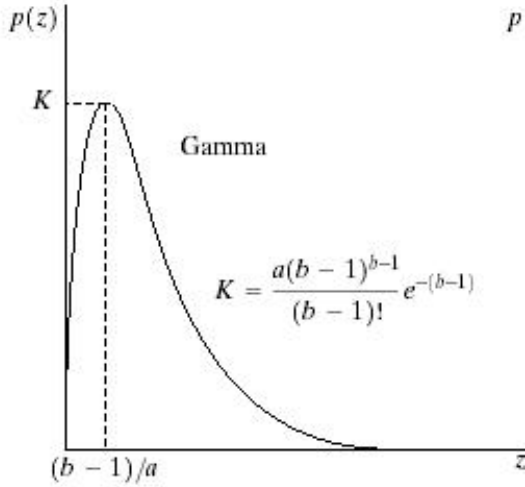
（4）脉冲噪声污染的的图像





9.1.4 图像噪声

（5）伽马噪声（爱尔兰噪声）



- 伽马噪声PDF：

$$p(z) = \begin{cases} \dfrac{a^b z^{b-1}}{(b-1)!} e^{-az} & z \geq 0 \\ 0 & z < 0 \end{cases}$$

- 其中，a>0，b为正整数且“！”表示阶乘。其密度的均值和方差为：

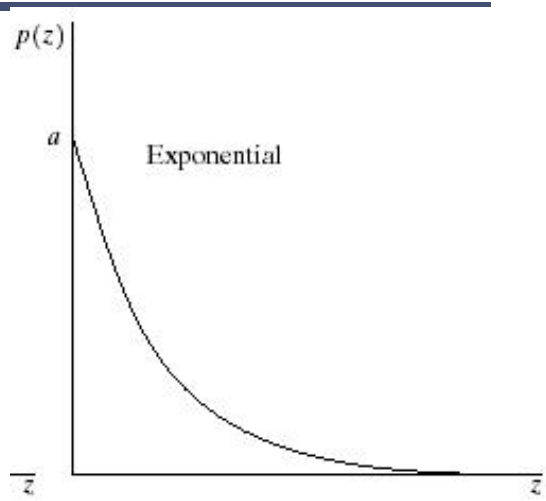
$$\mu = \frac{b}{a} \qquad \sigma^2 = \frac{b}{a^2}$$

9.1.4 图像噪声

（6）指数分布噪声

- 指数噪声的PDF：

$$p(z)=\begin{cases} ae^{-az} & z\geq 0 \\ 0 & z<0 \end{cases}$$



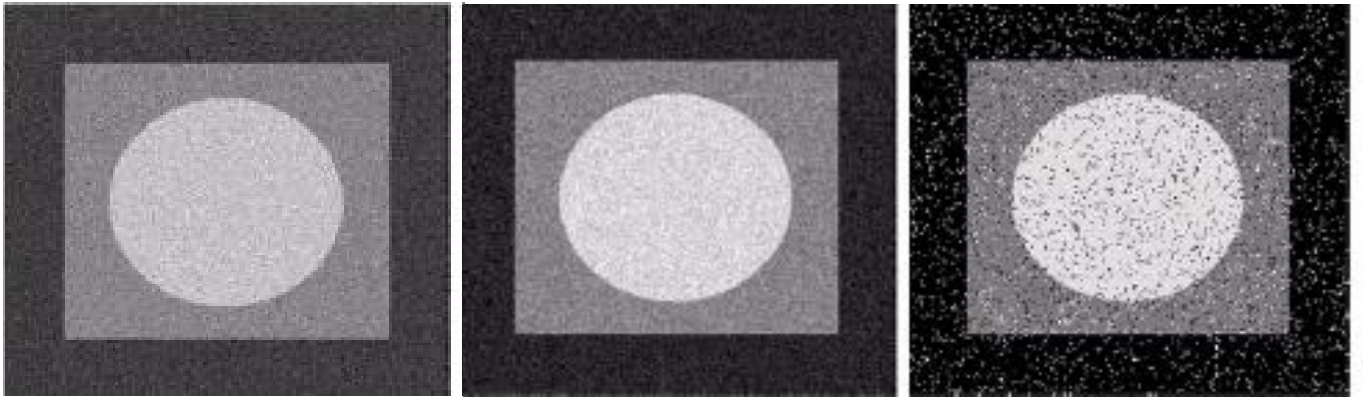
- 其中，a>0。概率密度函数的期望值和方差：

$$\mu = \frac{1}{a} \qquad \sigma^2 = \frac{1}{a^2}$$

- 注意，指数分布的概率密度函数是当b=1时爱尔兰概率分布的特殊情况。

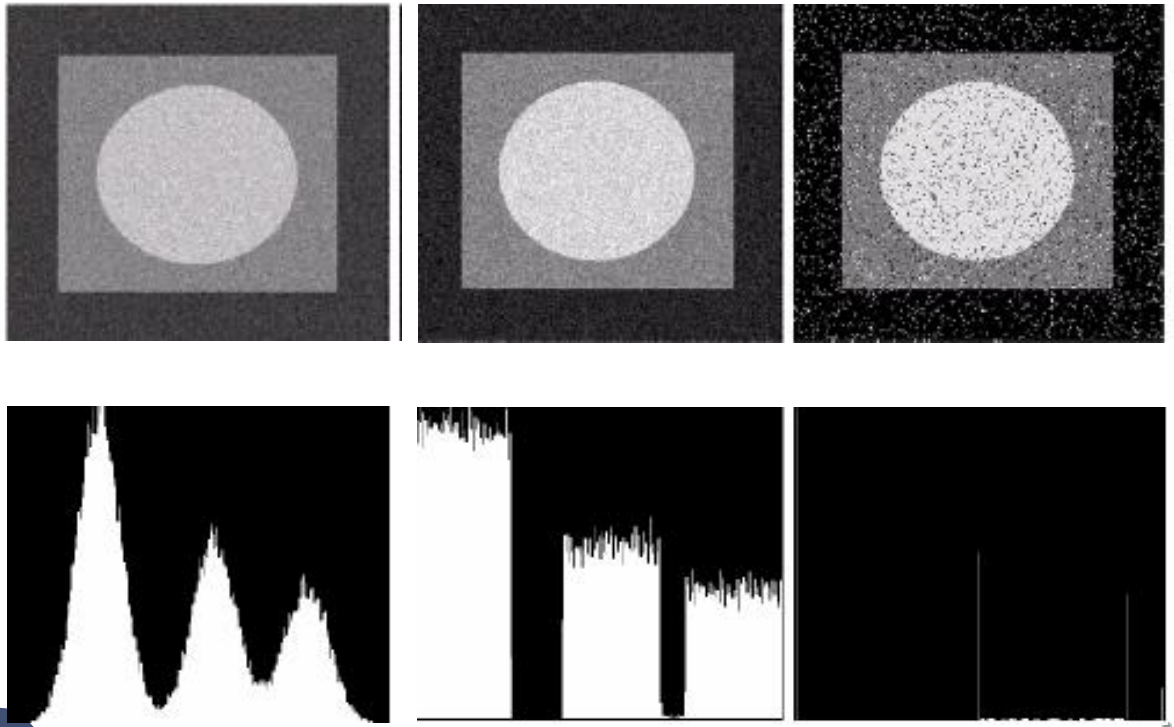
9.1.4 图像噪声

怎样判断一幅图像受某种噪声污染？

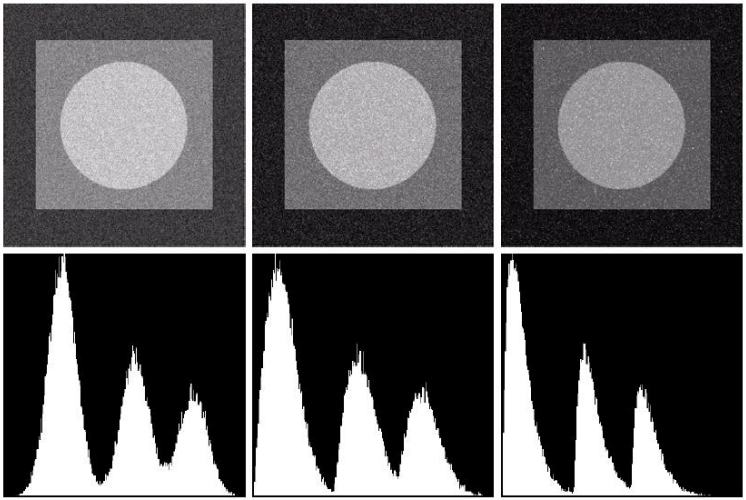


9.1.4 图像噪声

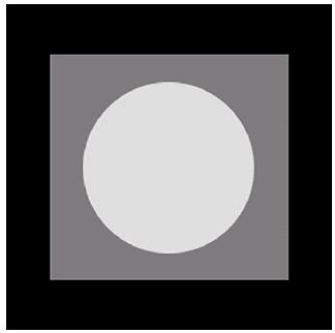
怎样判断一幅图像受某种噪声污染？

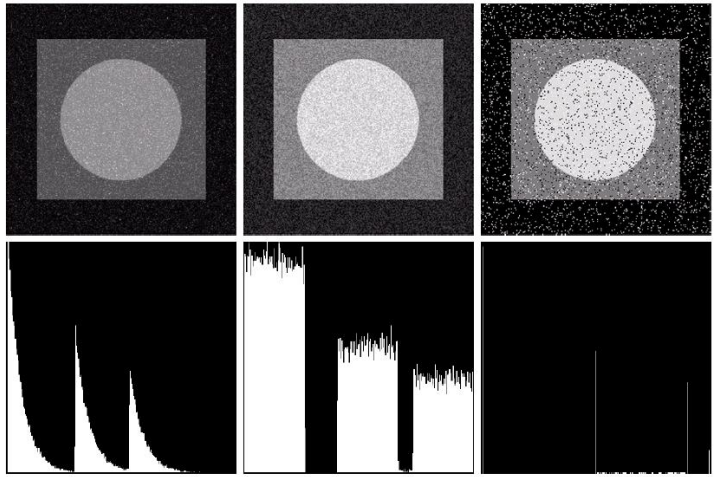


9.1.4 图像噪声



Gaussian　　Rayleigh　　Gamma





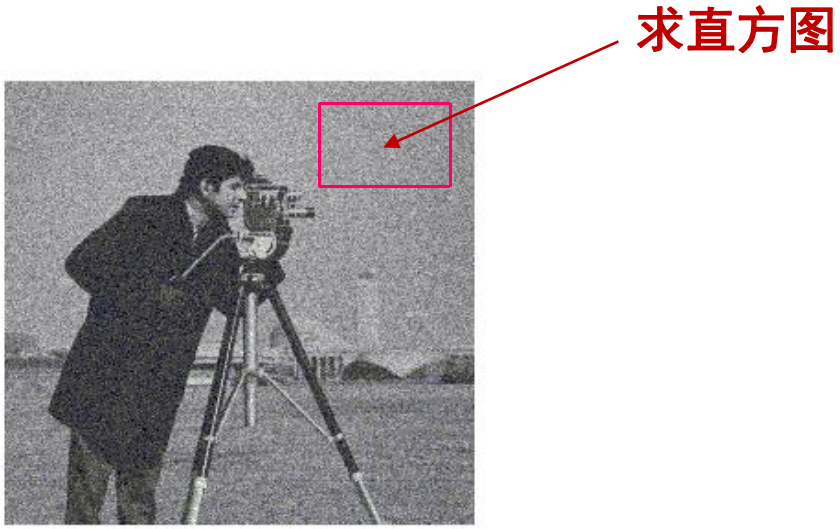
Exponential　　Uniform　　Salt & Pepper

样本噪声图像和它们的直方图

9.1.4 图像噪声

怎样判断一幅图像受某种噪声污染？

存在物体时，取一个图像块求它的直方图



9.2 图像退化函数的估计

9.2.1 基于模型的估计法

9.2.2 基于退化图像本身特性的估计法

9.2.1 基于模型的估计法

（1）定义

- 若已知引起退化的原因，根据基本原理推导出其退化模型，称为基于模型的估计法。

（2）运动模糊退化估计

- 运动模糊图像
  - 景物和摄像机之间的相对运动，曝光时间内，景物在不同时刻产生多个影像，叠加而导致的模糊，称为运动模糊

$$g(x,y)=\int_0^T f\left[x-x_0(t),y-y_0(t)\right]dt$$

$x_0(t)y_0(t)$ 为x、y方向上的运动分量，T为曝光时间

（2）运动模糊退化估计

- 运动模糊传递函数

$$H(u,v)=\int_0^T e^{-j2\pi[uat/T+vbt/T]}dt$$

$$=\frac{T}{\pi(ua+vb)}sin\left[\pi(ua+vb)\right]e^{-j\pi(ua+vb)}$$

匀速直线运动，T时间内，x、y方向上运动a和b

（2）运动模糊退化估计

- 运动模糊的点扩散函数

景物在x-y平面沿θ方向做匀速直线运动（θ是运动方向和x轴夹角），移动L个像素，点扩散函数为：

$$h(x,y)=\begin{cases}1/L & y=x\tan\theta,\ 0\le x\le L\cos\theta\\ 0 & y\ne x\tan\theta,\ -\infty<x<\infty\end{cases}$$

（2）运动模糊退化估计

- 例程

  设定运动方向和距离，对图像进行模糊处理

  根据点扩散函数设计运动模糊模板，并和原图像卷积，实现运动模糊效果，是MATLAB中fspecial函数实现运动模糊的设计思路。

（2）运动模糊退化估计

- 例程

原图

运动模糊图像

（2）运动模糊退化估计

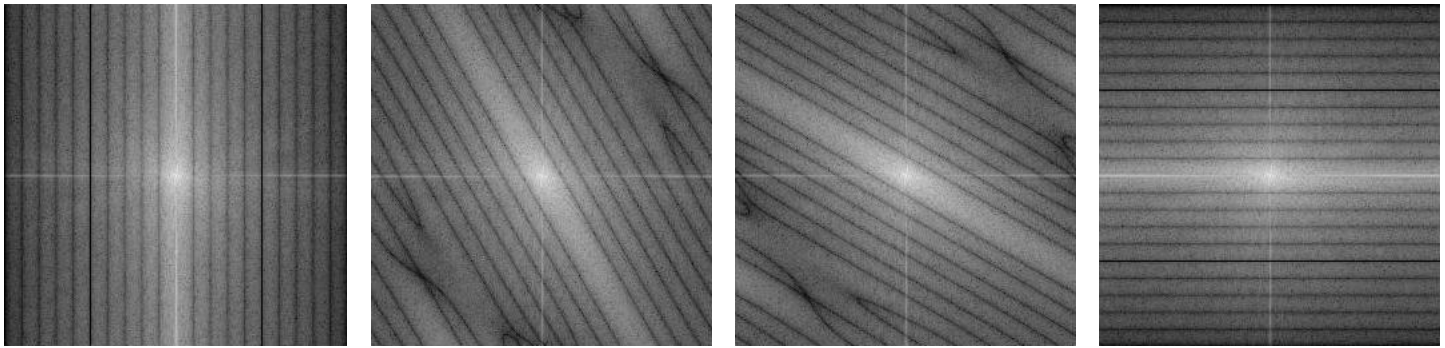
- 运动模糊点扩散函数参数估计
  - 基于频域特征的参数估计

    分析不同方向的运动模糊图像频谱变化

分别向0°、30°、60°、90°方向运动20个像素

（2）运动模糊退化估计

- 运动模糊点扩散函数参数估计

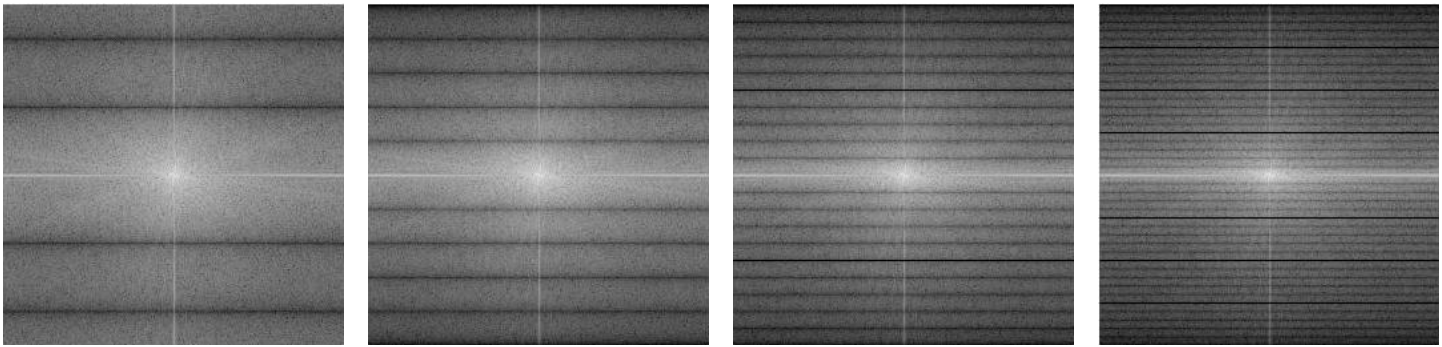


0°、30°、60°、90°方向运动模糊图像的频谱图

频谱图有黑色的平行条纹，方向总是与运动方向垂直，可以通过判定模糊图像频谱条纹的方向来确定实际的运动模糊方向

（2）运动模糊退化估计

- 运动模糊点扩散函数参数估计



90°方向上运动5、10、20、40个像素的模糊图像频谱图

图像频谱图条纹个数即为图像实际运动模糊的长度

（3）其他退化函数模型

- 散焦模糊退化函数

$$h(x,y)=\begin{cases}1/\pi R^2 & x^2+y^2\le R^2\\ 0 & \text{其他}\end{cases}$$

R为散焦半径

- 高斯退化函数

$$h(x,y)=\begin{cases}K\,exp\left[-\alpha\left(x^2+y^2\right)\right] & (x,y)\in S\\ 0 & \text{其他}\end{cases}$$

K为归一化常数，α为正常数

S为点扩散函数的圆形域

9.2.2 基于退化图像本身特性的估计法

对引起退化的物理性质不了解，或引起退化的过程过分复杂，无法用分析的方法确定点扩散函数，则可以采用退化图像本身的特性来估计

9.2.2 基于退化图像本身特性的估计法

估计h(x, y)或H(u, v)：估计退化函数

1. 大气湍流的退化函数

建模估计

$$H(u,v) = \exp[-c(u^2 + v^2)^{5/6}]$$

C：与湍流性质有关的常数

9.2.2 基于退化图像本身特性的估计法

a b
c d

FIGURE 5.25
Illustration of the atmospheric turbulence model. (a) Negligible turbulence. (b) Severe turbulence, $k = 0.0025$. (c) Mild turbulence, $k = 0.001$. (d) Low turbulence, $k = 0.00025$. (Original image courtesy of NASA.)

a. 原图；b. c. d. 不同k的退化图

9.2.2 基于退化图像本身特性的估计法

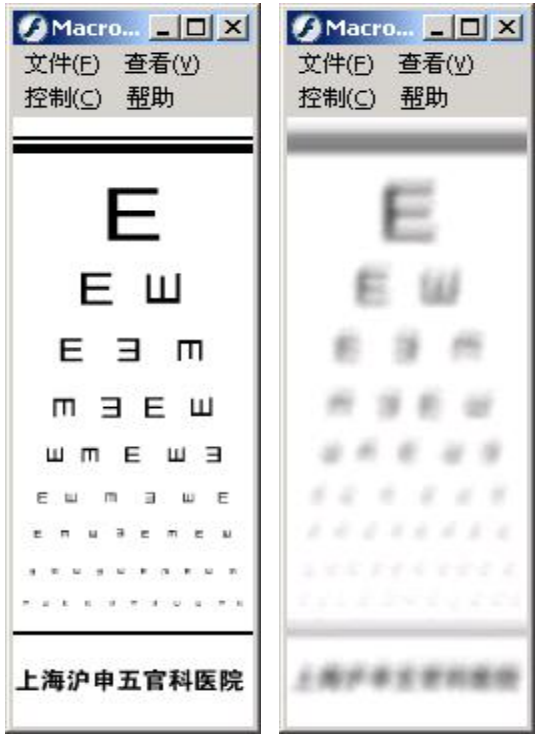
估计退化函数

2. 光学散焦

$$H(u,v) = J_1(\pi d\rho) \Big/ \pi d\rho$$

$$\rho = (u^2 + v^2)^{1/2}$$

d：是散焦点扩展函数的直径，$J_1(\bullet)$是第一类贝塞尔函数。



退化图

9.2.2 基于退化图像本身特性的估计法

9.3 图像复原的代数方法

根据退化模型，假设具备关于g、H、n的某些先验知识，确定某种最佳准则，寻找原图像f的最优估计

9.3.1 无约束最小二乘方复原

9.3.2 约束复原

9.3.1 无约束最小二乘方复原

- 噪声项：$n = g - Hf$
- 最佳准则：$J(\hat{f}) = \left\| g - H\hat{f} \right\|^2 = \left(g - H\hat{f} \right)^T \left(g - H\hat{f} \right)$

$J(\hat{f})$的最小值对应最优

$H\hat{f}$ 在最小二乘方意义上近似于g

选择 $\hat{f}$ 不受其他条件约束，称为无约束复原

- 最佳准则求最优：$\dfrac{\partial J(\hat{f})}{\partial \hat{f}} = -2H^T \left(g - H\hat{f} \right) = 0$

$$\hat{f} = H^{-1} \left(H^T \right)^{-1} H^T g = H^{-1} g$$

9.3.2 约束复原

附加某种约束条件，称为约束复原。有附加条件的极值问题可用拉格朗日乘数法来求解。

- 约束条件：$\|n\|^2 = \|g - H\hat{f}\|^2$
- 准则函数：$\|Q\hat{f}\|^2$ 最小为最优

Q为对原图像进行的某一线性运算

- 拉格朗日函数：$J(\hat{f}) = \|Q\hat{f}\|^2 + \lambda\left(\|g - H\hat{f}\|^2 - \|n\|^2\right)$

求极小值：$\hat{f} = \left(H^T H + \frac{1}{\lambda}Q^T Q\right)^{-1} H^T g$

9.4 典型图像复原方法

9.4.1 逆滤波复原

9.4.2 维纳滤波复原

9.4.3 等功率谱滤波

9.4.4 几何均值滤波

9.4.5 约束最小二乘方滤波

9.4.6 Richardson-Lucy算法

9.4.1 逆滤波复原

（1）原理

$$g(x,y)=f(x,y)*h(x,y)+n(x,y) \qquad G(u,v)=F(u,v)\cdot H(u,v)+N(u,v)$$

$$F(u,v)=\frac{G(u,v)}{H(u,v)}-\frac{N(u,v)}{H(u,v)} \qquad f(x,y)=\mathcal{F}^{-1}\left[F(u,v)\right]$$

$H(u,v)$不能为零，人为设置零点处取值

低通逆滤波 $$M(u,v)=\begin{cases}H^{-1}(u,v) & u^2+v^2\le D_0\\ 0 & u^2+v^2>D_0\end{cases}$$

9.4.1 逆滤波复原

（2）例程

对图像进行均值模糊，并进行逆滤波复原。

```
Image=im2double(rgb2gray(imread('flower.jpg')));
window=15;          [n,m]=size(Image);
n=n+window-1;     m=m+window-1;
h=fspecial('average',window);
BlurI=conv2(h,Image);
BlurandnoiseI=imnoise(BlurI,'salt & pepper',0.001);
figure,imshow(Image),title('Original Image');
figure,imshow(BlurI),title('Blurred Image');
figure,imshow(BlurandnoiseI)
                ,title('Blurred Image with noise');
```

9.4.1 逆滤波复原

```
h1=zeros(n,m);          h1(1:window,1:window)=h;
H=fftshift(fft2(h1));   H(abs(H)<0.0001)=0.01;
M=H.^(-1);              d0=sqrt(m^2+n^2)/20;
r1=floor(m/2);  r2=floor(n/2);
for u=1:m
   for v=1:n
      d=sqrt((u-r1)^2+(v-r2)^2);
      if d>d0
         M(v,u)=0;
      end
   end
end
```

9.4.1 逆滤波复原

（2）例程

```
G1=fftshift(fft2(BlurI));
G2=fftshift(fft2(BlurandnoiseI));
f1=ifft2(ifftshift(G1./H));
f2=ifft2(ifftshift(G2./H));
f3=ifft2(ifftshift(G2.*M));
result1=f1(1:n-window+1,1:m-window+1);
result2=f2(1:n-window+1,1:m-window+1);
result3=f3(1:n-window+1,1:m-window+1);
figure,imshow(abs(result1),[]),title('Filtered Image1');
figure,imshow(abs(result2),[]),title('Filtered Image2');
figure,imshow(abs(result3),[]),title('Filtered Image3');
```

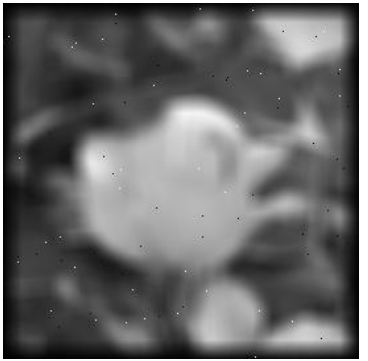
9.4.1 逆滤波复原

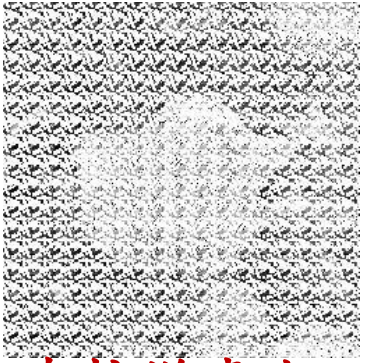
（2）例程

模糊图像

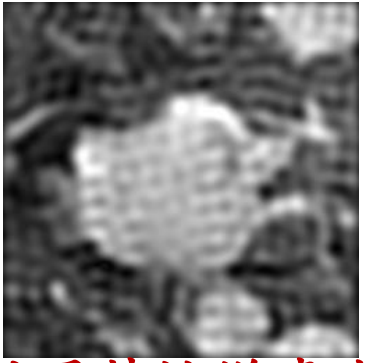
逆滤波

模糊加噪声





直接逆滤波



低通特性逆滤波

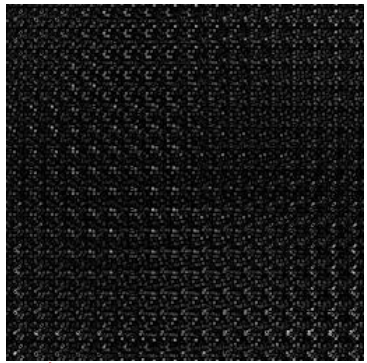
9.4.1 逆滤波复原

（2）例程

模糊图像

逆滤波

模糊加噪声



直接逆滤波

低通特性逆滤波

9.4.2 维纳滤波复原

（1）原理

- 一种有代表性的约束复原方法，使原始图像和复原图像之间**均方误差最小**的复原方法。

均方误差：$e^2 = E\left[\left(f - \hat{f}\right)^2\right]$

传递函数：$$H_w(u,v) = \frac{1}{H(u,v)} \cdot \frac{\left|H(u,v)\right|^2}{\left|H(u,v)\right|^2 + S_n(u,v)/S_f(u,v)}$$

功率谱：$S_f(u,v) = \left|F(u,v)\right|^2 \quad S_n(u,v) = \left|N(u,v)\right|^2$

噪声为零，噪声功率谱小，**维纳滤波即为逆滤波**；实际问题中，功率谱未知，用常数K来代替二者的比值

9.4.2 维纳滤波复原

（2）例程

对运动模糊的图像进行维纳滤波

- 函数

J = deconvwnr(I,PSF)　　J = deconvwnr(I,PSF,NSR)
J = deconvwnr(I,PSF,NCORR,ICORR)

- 程序

```
Image=im2double(rgb2gray(imread('flower.jpg')));
subplot(221),imshow(Image),title('Original Image');
LEN=21;　　THETA=11;
PSF=fspecial('motion', LEN, THETA);
BlurredI=imfilter(Image, PSF, 'conv', 'circular');
```

9.4.2 维纳滤波复原

（2）例程

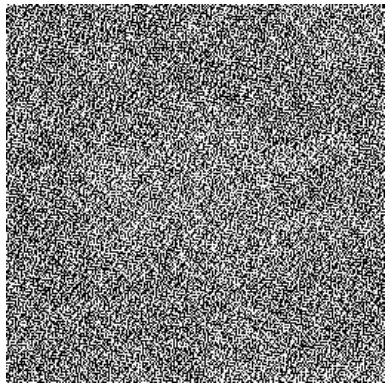
```
noise_mean = 0;         noise_var = 0.0001;
BlurandnoisyI=imnoise(BlurredI, 'gaussian',
                              noise_mean, noise_var);
subplot(222), imshow(BlurandnoisyI),
                 title('Simulate Blur and Noise');
estimated_nsr = 0;
result1= deconvwnr(BlurandnoisyI, PSF, estimated_nsr);
subplot(223),imshow(result1),
                 title('Restoration Using NSR = 0');
estimated_nsr = noise_var / var(Image(:));
result2 = deconvwnr(BlurandnoisyI, PSF, estimated_nsr);
subplot(224),imshow(result2),
                 title('Restoration Using Estimated NSR');
```

9.4.2 维纳滤波复原

（2）例程

- 效果

运动模糊加高斯噪声图像



维纳滤波复原（NSR=0）

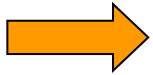
维纳滤波复原（估计NSR）

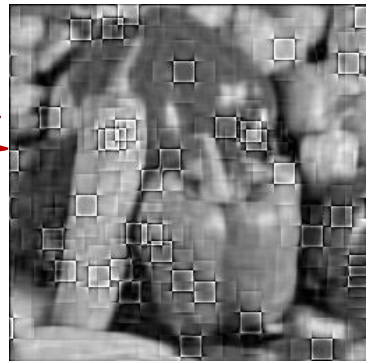
9.4.2 维纳滤波复原

（2）例程

模糊加噪声图像

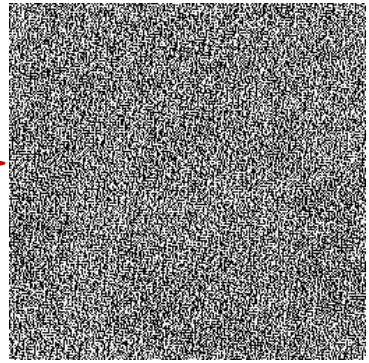
维纳滤波





运动模糊加高斯噪声

维纳滤波



NSR=0

估计NSR

9.4.3 等功率谱滤波

（1）原理

- 使原始图像和复原图像功率谱相等的复原方法。

  设图像和噪声均属于均匀随机场，噪声均值为零，且与图像不相关。

$$S_g(u,v)=|H(u,v)|^2 S_f(u,v)+S_n(u,v) \quad S_{\hat{f}}(u,v)=S_g(u,v)|M(u,v)|^2$$

$$\because S_{\hat{f}}(u,v)=S_f(u,v)$$

传递函数：$$M(u,v)=\left[\frac{1}{|H(u,v)|^2+S_n(u,v)/S_f(u,v)}\right]^{1/2}$$

无噪声时，为逆滤波；用常数K来代替功率谱比值

9.4.3 等功率谱滤波

（2）例程

对运动模糊加噪声图像进行等功率谱滤波复原

```
Image=im2double(rgb2gray(imread('flower.jpg')));
[n,m]=size(Image);
figure,imshow(Image),title('Original Image');

LEN=21;    THETA=11;
PSF=fspecial('motion', LEN, THETA);
BlurredI=conv2(PSF,Image);
figure,imshow(BlurredI),title('motion blur');
```

9.4.3 等功率谱滤波

（2）例程

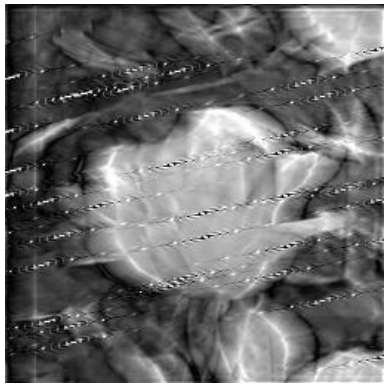
```
[nh,mh]=size(PSF);
n=n+nh-1;          m=m+mh-1;
noise=imnoise(zeros(n,m),'salt & pepper',0.001);
BlurandnoiseI=BlurredI+noise;
figure,imshow(BlurandnoiseI),
        title('Blurred Image with noise');

h1=zeros(n,m);      h1(1:nh,1:mh)=PSF;
H=fftshift(fft2(h1));
K=sum(noise(:).^2)/sum(Image(:).^2);
M=(1./(abs(H).^2+K)).^0.5;
```

9.4.3 等功率谱滤波

（2）例程

```
G=fftshift(fft2(BlurandnoiseI));
f=ifft2(ifftshift(G.*M));
result=f(1:n-nh+1,1:m-mh+1);
figure,imshow(abs(result)),title('Filtered Image');
```

运动模糊加椒盐噪声



等功率谱滤波

9.4.4 几何均值滤波

$$M(u,v)=\left[\frac{H^*(u,v)}{|H(u,v)|^2}\right]^{\alpha}\left[\frac{H^*(u,v)}{|H(u,v)|^2+\gamma S_n(u,v)/S_f(u,v)}\right]^{1-\alpha}$$

$\alpha=1$ 逆滤波器

$\alpha=0$ 参数化的维纳滤波器

$\alpha=1/2,\gamma=1$ 等功率谱滤波器

……

通过灵活选择 α,γ 的值来获得良好的平滑效果

9.4.5 约束最小二乘方滤波

（1）原理　采用最小化原图二阶微分的方法

Q为拉普拉斯算子：$l(x,y)=\begin{pmatrix}0 & 1 & 0\\ 1 & -4 & 1\\ 0 & 1 & 0\end{pmatrix}$

最优化准则为：$min\left(f(x,y)*l(x,y)\right)$

$$\hat{f}=\left(H^TH+\frac{1}{\lambda}L^TL\right)^{-1}H^Tg$$

$$\hat{F}(u,v)=\left[\frac{H_e^*(u,v)}{\left|H_e(u,v)\right|^2+\frac{1}{\lambda}\left|L_e(u,v)\right|^2}\right]G_e(u,v)$$

9.4.5 约束最小二乘方滤波

（2）实现

根据求解公式，通过调整参数使得 $\|e\|^2 = \|n\|^2$

$$E(u,v) = G(u,v) - H(u,v)\hat{F}(u,v)$$

$$\|e\|^2 = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} e^2(x,y)$$

$$\sigma_n^2 = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left[n(x,y) - \mu_n\right]^2$$

$$\mu_n = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} n(x,y)$$

$$\|n\|^2 = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} n^2(x,y) = MN\left[\sigma_n^2 + \mu_n^2\right]$$

9.4.5 约束最小二乘方滤波

（3）例程　对模糊的图像进行约束最小二乘方滤波

```
Image=im2double(rgb2gray(imread('flower.jpg')));
window=15;                    [N,M]=size(Image);
N=N+window-1;                 M=M+window-1;
h=fspecial('average',window);
BlurI=conv2(h,Image);
sigma=0.001;                  miun=0;
nn=M*N*(sigma+miun*miun);
BlurandnoiseI=imnoise(BlurI,'gaussian',miun,sigma);
figure,imshow(BlurandnoiseI),
        title('Blurred Image with noise');
```

9.4.5 约束最小二乘方滤波

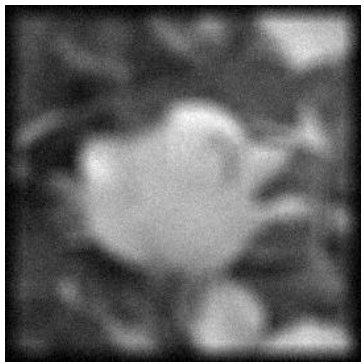
（3）例程

```
h1=zeros(N,M);
h1(1:window,1:window)=h;
H=fftshift(fft2(h1));

lap=[0 1 0;1 -4 1;0 1 0];
L=zeros(N,M);
L(1:3,1:3)=lap;
L=fftshift(fft2(L));

G=fftshift(fft2(BlurandnoiseI));
gama=0.3;    step=0.01;
alpha=nn*0.001;    flag=true;
while flag
   MH=conj(H)./(abs(H).^2+
        gama*(abs(L).^2));
   F=G.*MH;     E=G-H.*F;
   E=abs(ifft2(ifftshift(E)));
   ee=sum(E(:).^2);
   if ee<nn-alpha
      gama=gama+step;
   elseif ee>nn+alpha
      gama=gama-step;
   else
      flag=false;
   end
end
```

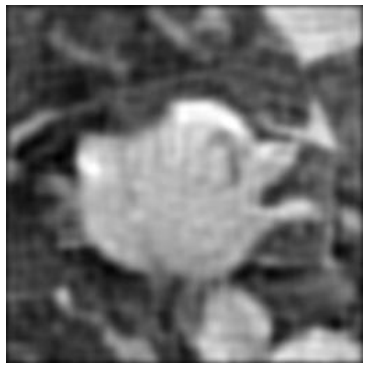
9.4.5 约束最小二乘方滤波

（3）例程

```
MH=conj(H)./(abs(H).^2+gama*(abs(L).^2));
f=ifft2(ifftshift(G.*MH));
result=f(1:N-window+1,1:M-window+1);
[J, LAGRA]=deconvreg(BlurandnoiseI,h,nn);
figure,imshow(J,[]);
figure,imshow(abs(result),[]),title('Filtered Image');
```



模糊加高斯噪声



约束最小二乘滤波

9.4.5 约束最小二乘方滤波

（3）例程

模糊加高斯噪声

约束最小

二乘滤波

9.4.5 约束最小二乘方滤波

（3）例程

- 函数

```
J = deconvreg(I,PSF,NP);
J = deconvreg(I,PSF,NP,LRANGE);
J = deconvreg(I,PSF,NP,LRANGE,REGOP);
[J, LAGRA] = deconvreg(I,PSF,...)。
```

- 程序

接前程序中获取模糊图像BlurrednoisyI一段儿

```
J=deconvreg(BlurrednoisyI,h,nn);
figure,imshow(J,[]);
```

9.4.6 Richardson–Lucy算法

（1）原理

- 简称RL算法，图像复原的经典算法之一，因William Richardson和Leon Lucy各自独立提出而得名。
- 算法假设图像服从泊松分布，采用最大似然法得到估计原始图像信息的迭代表达式：

$$\hat{f}_{k+1}(x,y)=\hat{f}_k(x,y)\left[h(-x,-y)*\frac{g(x,y)}{h(x,y)*\hat{f}_k(x,y)}\right]$$

k表示迭代次数

9.4.6 Richardson–Lucy算法

（2）例程

对模糊的图像进行RL算法复原

- 函数

```
J=deconvlucy(I,PSF);
J=deconvlucy(I,PSF,NUMIT);
J=deconvlucy(I,PSF,NUMIT,DAMPAR);
J=deconvlucy(I,PSF,NUMIT,DAMPAR,WEIGHT);
J=deconvlucy(I,PSF,NUMIT,DAMPAR,WEIGHT
                                ,READOUT)。
```

9.4.6 Richardson–Lucy算法

（2）例程

- 程序

```
I=im2double(rgb2gray(imread('flower.jpg')));
figure,imshow(I),title('原图像');
PSF=fspecial('gaussian',7,10);
V=0.0001;
IF1=imfilter(I,PSF);
BlurredNoisy=imnoise(IF1,'gaussian',0,V);
figure,imshow(BlurredNoisy),title('高斯模糊加噪声图像');
```

9.4.6 Richardson–Lucy算法

（2）例程

```
WT=zeros(size(I));%产生权重矩阵
WT(5:end-1,5:end-4)=1;
%使用不同的参数进行复原
J1=deconvlucy(BlurredNoisy,PSF);
J2=deconvlucy(BlurredNoisy,PSF,50,sqrt(V));
J3=deconvlucy(BlurredNoisy,PSF,100,sqrt(V),WT);
figure,imshow(J1),title('10次迭代');
figure,imshow(J2),title('50次迭代');
figure,imshow(J3),title('100次迭代');
```

9.4.6 Richardson–Lucy算法

（2）例程

高斯模糊
加噪声

10次迭代
去模糊

50次迭代
去模糊

100次迭代
去模糊

9.5 盲去卷积复原

- 不以PSF知识为基础的图像复原方法

（1）最大似然估计的盲图像复原算法

在PSF未知的情况下，根据退化图像、原始图像以及PSF的一些先验知识，采用概率理论建立似然函数，再对似然函数求最大值，实现原始图像和PSF的估计重建。

（2）原理

设退化图像$g(x,y)$的概率为$P(g)$，原始图像$f(x,y)$的概率为$P(f)$，由$f(x,y)*h(x,y)$估计$g(x,y)$的概率为$P(g|h*f)$，由$g(x,y)$估计$f(x,y)*h(x,y)$的概率为$P(h*f|g)$，由贝叶斯定理可知：

$$P(h*f|g)=\frac{P(g|h*f)P(f)P(h)}{P(g)}$$

最大时，认为原始图像和PSF最大概率逼近真实结果，即最大程度实现了原始图像和PSF的估计重建。

（2）原理

- 代价函数J:

$$J(h,f)=-ln\left[P(h*f|g)\right]=-ln\left[P(g|h*f)\right]-ln\left[P(f)\right]-ln\left[P(h)\right]$$

代价函数取最小值对应最优结果

（3）例程

对模糊的图像进行最大似然估计盲复原滤波

- 函数

```
[J,PSF] = deconvblind(I,INITPSF);
[J,PSF] = deconvblind(I,INITPSF,NUMIT);
[J,PSF] = deconvblind(I,INITPSF,NUMIT,DAMPAR);
[J,PSF] = deconvblind(I,INITPSF,NUMIT,DAMPAR
                                        ,WEIGHT);
[J,PSF] = deconvblind(I,INITPSF,NUMIT,DAMPAR
                                  ,WEIGHT,READOUT)。
```

（3）例程

- 程序

```
I=im2double(rgb2gray(imread('flower.jpg')));
PSF=fspecial('gaussian',7,10);
V=0.0001;                    IF1=imfilter(I,PSF);
BlurredNoisy=imnoise(IF1,'gaussian',0,V);
WT = zeros(size(I));     WT(5:end-4,5:end-4) = 1;
INITPSF = ones(size(PSF));
[J,P] = deconvblind(BlurredNoisy,INITPSF,20,10*sqrt(V),WT);
subplot(221),imshow(BlurredNoisy),title('高斯模糊加噪声图像');
subplot(222),imshow(PSF,[]),title('真正的PSF');
subplot(223),imshow(J),title('盲复原图像');
subplot(224),imshow(P,[]),title('重建的PSF');
```

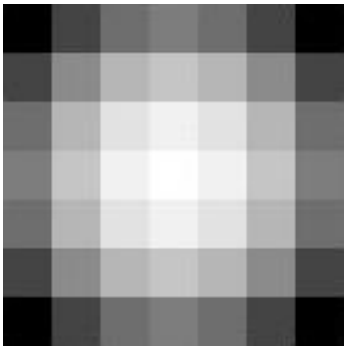
9.5 盲去卷积复原

（3）例程

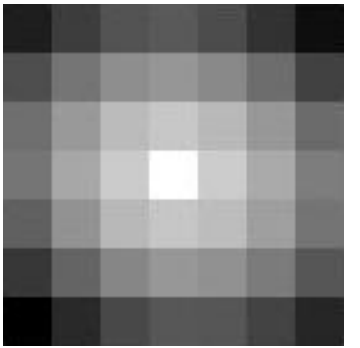
- 效果

高斯模糊加噪声

盲复原图像



真正的PSF



重建的PSF

9.6 几何失真校正

（1）原理

- 在图像生成和显示的过程中，由于成像系统本身具有的非线性，或者拍摄时成像系统光轴和景物之间存在一定倾斜角度，往往会造成图像的几何失真（几何畸变），这也是一种图像退化。
- 几何失真的校正：通过几何变换来校正失真图像中像素的位置，以便恢复原来像素空间关系的复原技术。

关键在于变换前后点的空间关系

9.6 几何失真校正

（1）原理

原图像：$f(x,y)$ 几何失真图像：$g(x',y')$

几何失真前后像素点的坐标满足：$\begin{cases} x' = h_1(x,y) \\ y' = h_2(x,y) \end{cases}$

设几何失真是线性的变换：$\begin{cases} x' = ax + by + c \\ y' = dx + ey + f \end{cases}$

几何失真校正需计算6个系数，在失真前后的图像中确定三个对应点，则可以通过组成方程组求解

$$\begin{cases} x_1' = ax_1 + by_1 + c & y_1' = dx_1 + ey_1 + f \\ x_2' = ax_2 + by_2 + c & y_2' = dx_2 + ey_2 + f \\ x_3' = ax_3 + by_3 + c & y_3' = dx_3 + ey_3 + f \end{cases}$$

（2）例程

产生几何失真图像，并利用交互式选择连接点工具选择连接点。

- 函数

T = maketform(TRANSFORMTYPE,...)

B = imtransform(A,TFORM,INTERP,
PARAM1,VAL1,PARAM2,VAL2,...)

cpselect(INPUT,BASE)：启动交互选择连接点工具

TFORM = cp2tform(INPUT_POINTS,BASE_POINTS
,TRANSFORMTYPE)：根据连接点建立几何变换结构

（2）例程

- 程序

```
Image=im2double(imread('lotus.jpg'));
[h,w,c]=size(Image);
figure,imshow(Image),title('原图');
RI=imrotate(Image,20);
tform=maketform('affine',[1 0.5 0;0.5 1 0; 0 0 1]);
NewImage=imtransform(RI,tform);
figure,imshow(NewImage),title('几何畸变的图像');
```

（2）例程

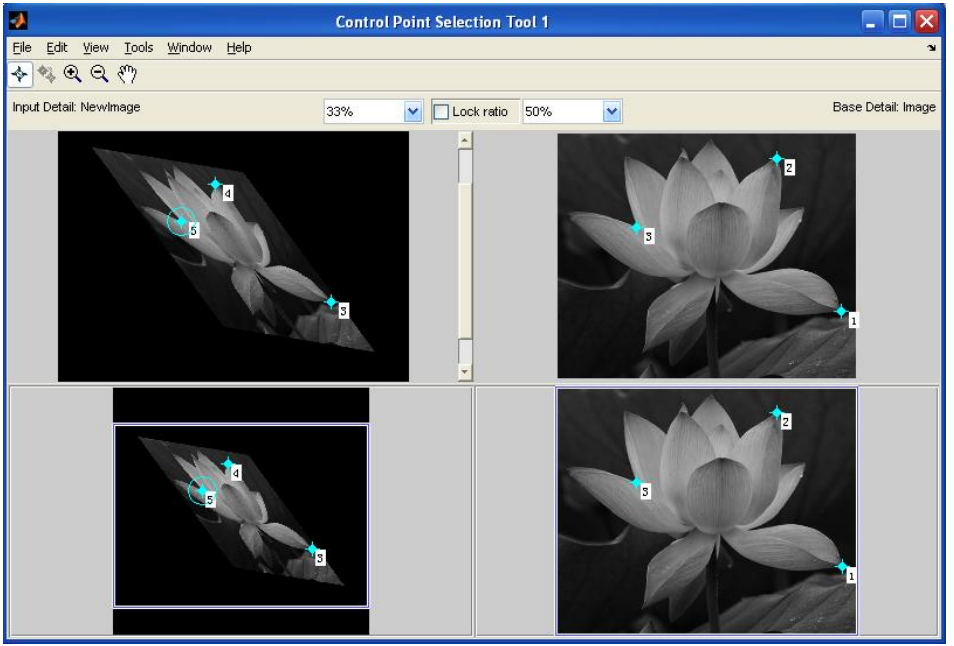
- 程序

```
cpselect(NewImage,Image);
input_points=[709 577;409 270;320 370];
base_points=[487 305;374 41;134 159];
tform=cp2tform(input_points,base_points,'affine');
result=imtransform(NewImage,tform,
                        'XData',[1 w],'YData',[1 h]);
figure,imshow(result),title('校正后的图像');
```

9.6 几何失真校正

（2）例程

- 效果



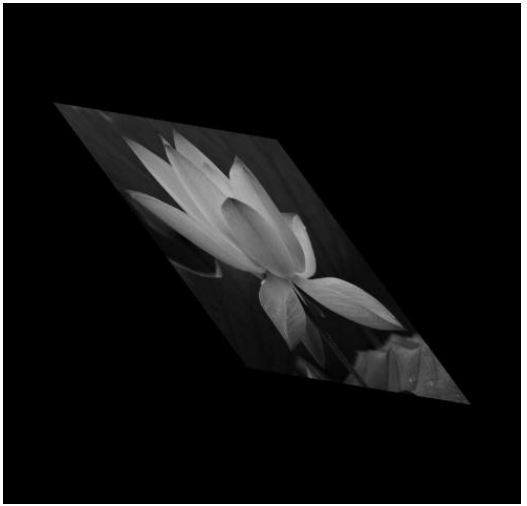
连接点选择

9.6 几何失真校正

（2）例程

- 效果

原始图像



几何失真图像

校正后的图像

（2）例程

- 分析

例程是采用几何变换生成的几何失真图像，手工标记的对应点，在实际问题中，需要检测图中的特殊点来建立对应关系。

思考

- 9.1简述图像退化的基本模型，并写出离散退化模型。
- 9.2简述什么是约束复原，什么是无约束复原。
- 9.3简述逆滤波复原的基本原理以及存在的问题。
- 9.4简述维纳滤波的原理。
- 9.5一幅退化图像，不知道原图像的功率谱，仅知道噪声的方差，请问采用何种方法复原图像较好？为什么？
- 9.6查找资料，了解图像复原技术的扩展应用及其核心技术。

编程实践

- 9.6编写程序，对一幅灰度图像进行运动模糊，尝试实现基于频域特征的运动模糊参数估计。
- 9.7编写程序，对一幅灰度图像进行高斯模糊并叠加高斯噪声，设计逆滤波器、维纳滤波器和约束最小二乘方滤波器对其进行复原，并比较复原效果。
- 9.8编写程序，对一幅灰度图像进行运动模糊并叠加噪声，设计几何均值滤波器，改变参数，观察复原效果。
- 9.9编写程序，打开一幅灰度图像，进行组合几何变换，并对其进行几何校正。