



第8章 图像复原

主讲教师 张涛

电子信息与电气工程



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。



- 8.1 图像退化模型
- 8.2 图像退化函数的估计
- 8.3 图像复原的代数方法
- 8.4 典型图像复原方法
- 8.5 盲去卷积复原
- 8.6 几何失真校正



8.1 图像退化模型

8.1.1 连续退化模型

8.1.2 离散退化模型

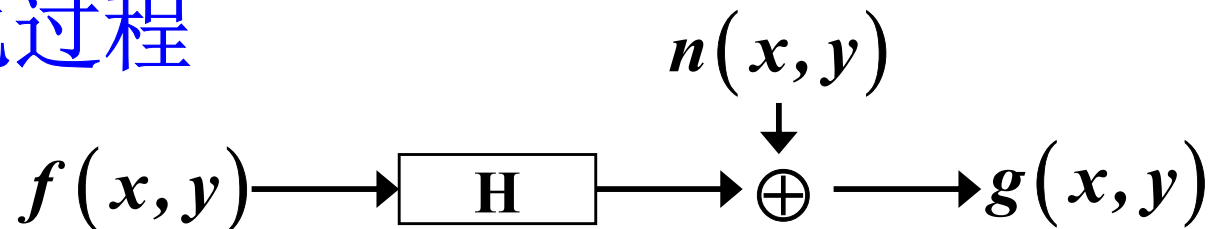
8.1.3 图像复原



8.1.1 连续退化模型

图像退化模型

(1) 退化过程



抽象为一个退化系统 H 以及加性噪声的影响

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

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

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



8.1.1 连续退化模型

图像退化模型

(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)



8.1.2 离散退化模型

图像退化模型

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

■ 采样延拓

$$f_e(x, y) = \begin{cases} f(x, y) & 0 \leq x \leq A-1, 0 \leq y \leq B-1 \\ 0 & A \leq x \leq M-1, B \leq y \leq N-1 \end{cases}$$

$$h_e(x, y) = \begin{cases} h(x, y) & 0 \leq x \leq C-1, 0 \leq y \leq D-1 \\ 0 & C \leq x \leq M-1, D \leq y \leq N-1 \end{cases}$$

■ 二维离散卷积退化模型

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



8.1.3 图像复原

图像退化模型

(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 $f(x,y)$



8.1.3 图像复原

图像退化模型

(2) 步骤

- 确定图像的退化函数

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

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



8.2 图像退化函数的估计

8.2.1 基于模型的估计法

8.2.2 基于退化图像本身特性的估计法



8.2.1 基于模型的估计法

图像退化函数的估计

(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。



8.2.1 基于模型的估计法

图像退化函数的估计

(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

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

$x_0(t)y_0(t)$ 为 x 、 y 方向上的运动分量， T 为曝光时间



8.2.1 基于模型的估计法

图像退化函数的估计

(2) 运动模糊退化估计

■ 运动模糊传递函数

$$\begin{aligned} H(u, v) &= \int_0^T e^{-j2\pi[uat/T + vbt/T]} dt \\ &= \frac{T}{\pi(ua + vb)} \sin[\pi(ua + vb)] e^{-j\pi(ua + vb)} \end{aligned}$$

匀速直线运动， T 时间内， x 、 y 方向上运动 a 和 b



8.2.1 基于模型的估计法

图像退化函数的估计

(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：

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



8.2.1 基于模型的估计法

图像退化函数的估计

(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。



8.2.1 基于模型的估计法

图像退化函数的估计

(2) 运动模糊退化估计

■ 例程



原图



运动模糊图像

8.2.1 基于模型的估计法

图像退化函数的估计

(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

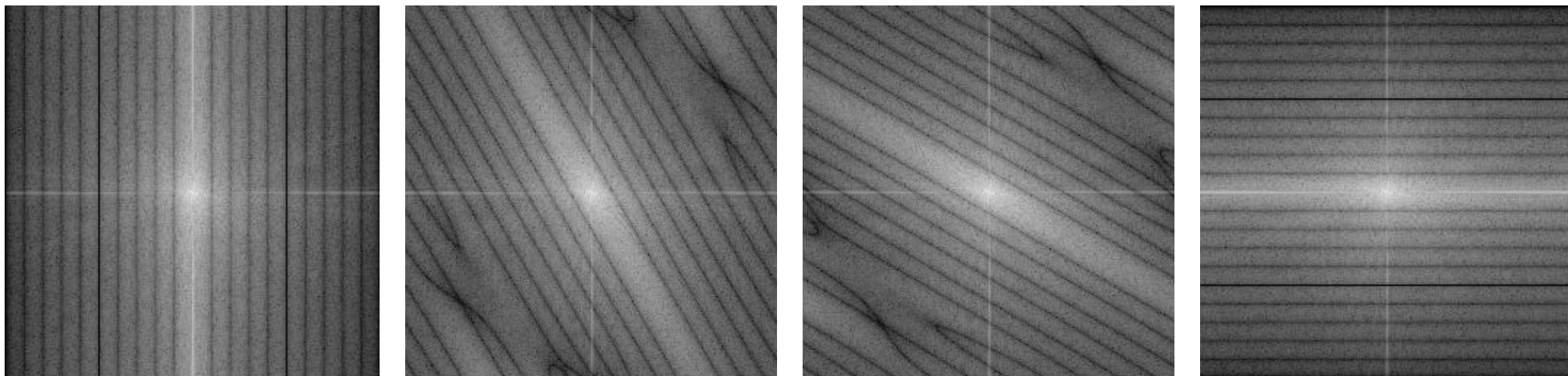


8.2.1 基于模型的估计法

图像退化函数的估计

(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

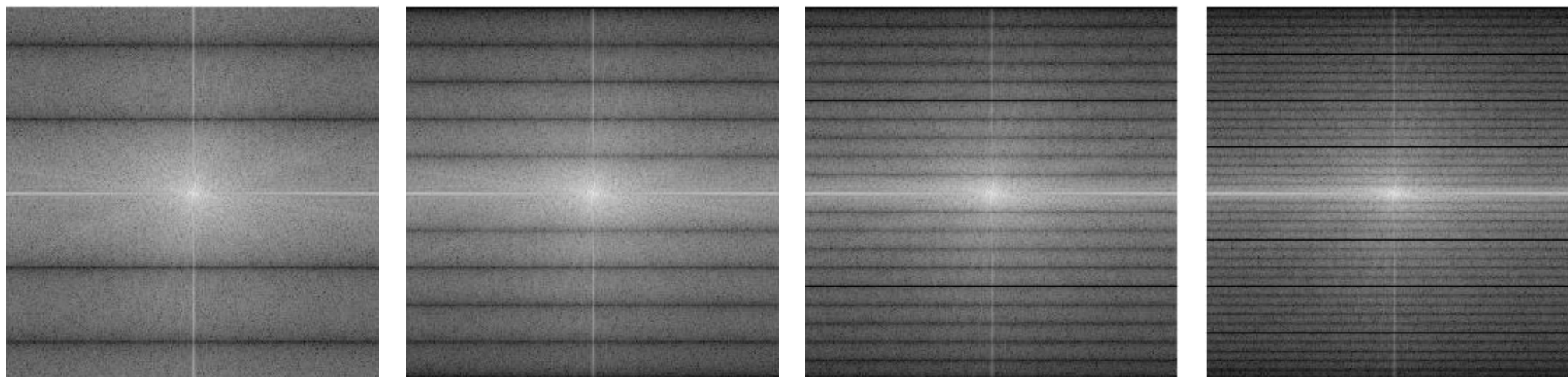


8.2.1 基于模型的估计法

图像退化函数的估计

(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



8.2.1 基于模型的估计法

图像退化函数的估计

(3) 其他退化函数模型

■ 散焦模糊退化函数

$$h(x, y) = \begin{cases} 1/\pi R^2 & x^2 + y^2 \leq R^2 \\ 0 & \text{其他} \end{cases} \quad \begin{matrix} R \text{为散焦半} \\ \text{径} \end{matrix}$$

■ 高斯退化函数

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

K 为归一化常数, α 为正常数

S 为点扩散函数的圆形域



8.2.2 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



8.3 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

8.3.1 无约束最小二乘方复原

8.3.2 约束复原



8.3.1 无约束最小二乘方复原

图像复原的代数方法

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

$J(\hat{f})$ 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: $\frac{\partial J(\hat{f})}{\partial \hat{f}} = -2H^T (g - H\hat{f}) = 0$

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



8.3.2 约束复原

图像复原的代数方法

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: $J(\hat{f}) = \|Q\hat{f}\|^2 + \lambda(\|g - H\hat{f}\|^2 - \|n\|^2)$

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



8.4 典型图像复原方法

8.4.1 逆滤波复原

8.4.2 维纳滤波复原

8.4.3 等功率谱滤波

8.4.4 几何均值滤波

8.4.5 约束最小二乘方滤波

8.4.6 Richardson-Lucy算法



8.4.1 逆滤波复原

典型图像复原方法

(1) 原理

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

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

$H(u, v)$ 不能为零，人为设置零点处取值

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



8.4.1 逆滤波复原

典型图像复原方法

(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');
```



8.4.1 逆滤波复原

典型图像复原方法

```
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
end
```



8.4.1 逆滤波复原

典型图像复原方法

(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');
```



8.4.1 逆滤波复原

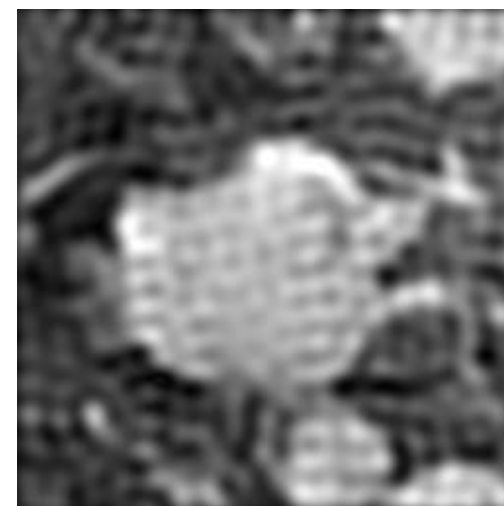
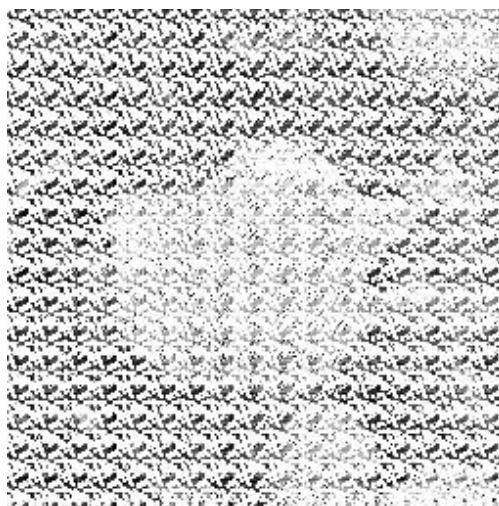
典型图像复原方法

(2) 例程

模糊图像



模糊加噪声



直接逆滤波

低通特性逆滤波



8.4.1 逆滤波复原

典型图像复原方法

(2) 例程

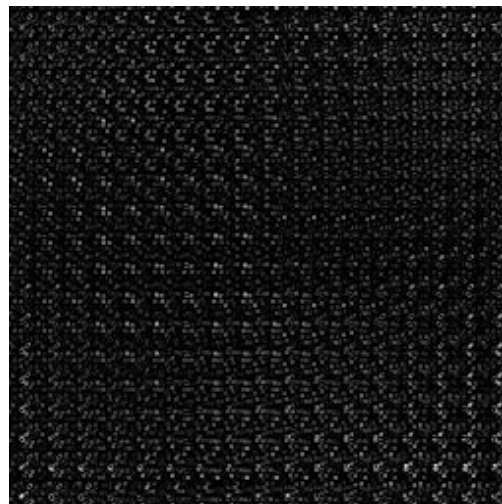
模糊图像



逆滤波



模糊加噪声



直接逆滤波 低通特性逆滤波



8.4.2 维纳滤波复原

典型图像复原方法

(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。

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

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

$$\text{功率谱: } S_f(u, v) = |F(u, v)|^2 \quad S_n(u, v) = |N(u, v)|^2$$

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



8.4.2 维纳滤波复原

典型图像复原方法

(2) 例程

对运动模糊的图像进行维纳滤波

■ 函数

$J = \text{deconvwnr}(I, \text{PSF})$ $J = \text{deconvwnr}(I, \text{PSF}, \text{NSR})$

$J = \text{deconvwnr}(I, \text{PSF}, \text{NCORR}, \text{ICORR})$

■ 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');
```



8.4.2 维纳滤波复原

典型图像复原方法

(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');
```

8.4.2 维纳滤波复原

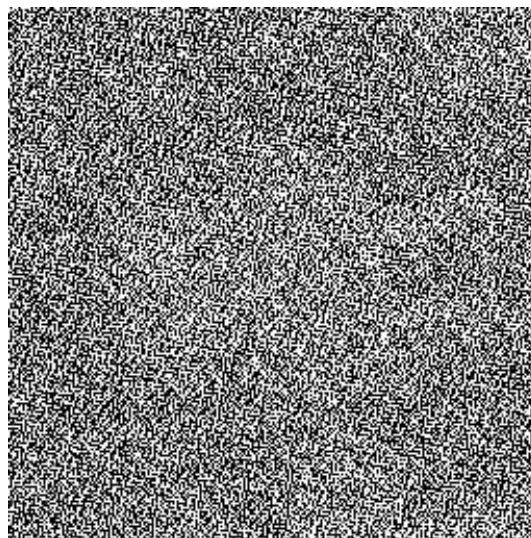
典型图像复原方法

(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)



8.4.2 维纳滤波复原

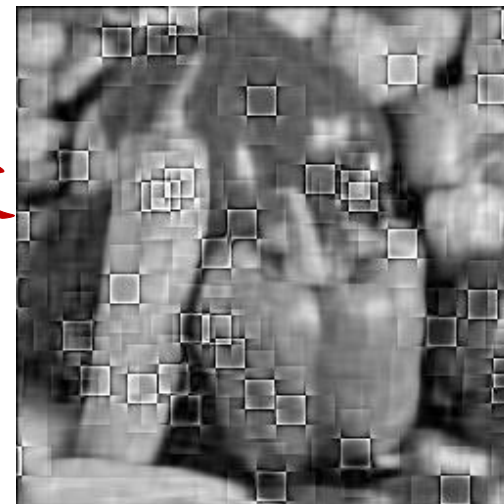
典型图像复原方法

(2) 例程

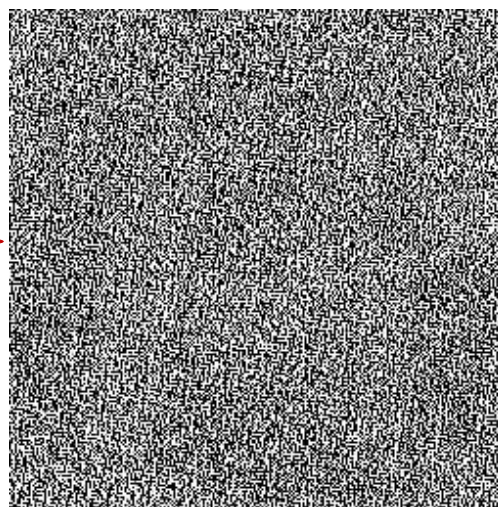
模糊加噪声图像



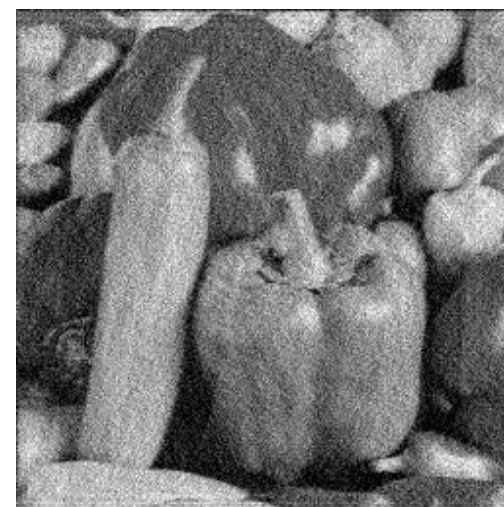
维纳滤波



维纳滤波



NSR=0



估计NSR

运动模糊加高斯噪声





8.4.3 等功率谱滤波

典型图像复原方法

(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。

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

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

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

无噪声时，为逆滤波；用常数K来代替功率谱比值



8.4.3 等功率谱滤波

典型图像复原方法

(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');
```



8.4.3 等功率谱滤波

典型图像复原方法

(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;
```


8.4.3 等功率谱滤波

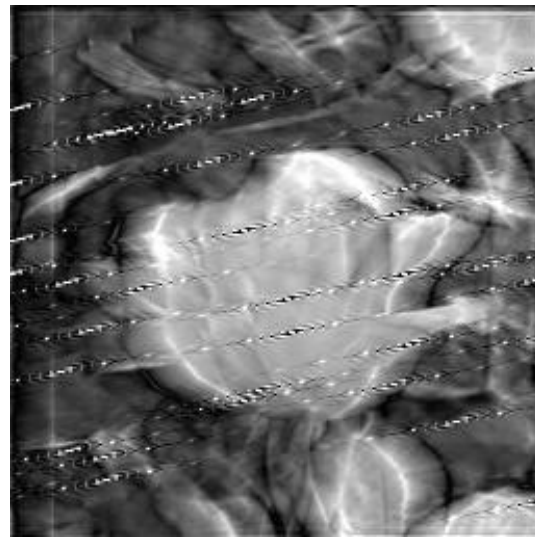
典型图像复原方法

(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



8.4.4 几何均值滤波

典型图像复原方法

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

$\alpha = 1$ 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



8.4.5 约束最小二乘方滤波

典型图像复原方法

(1) 原理 采用最小化原图二阶微分的方法

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

最优化准则为: $\min(f(x, y) * l(x, y))$

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

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



8.4.5 约束最小二乘方滤波

典型图像复原方法

(2) 实现

根据求解公式，通过调整参数使得 $\|e\|^2 = \|n\|^2$

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

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

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

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

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



8.4.5 约束最小二乘方滤波

典型图像复原方法

(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');
```



8.4.5 约束最小二乘方滤波

典型图像复原方法

(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
```

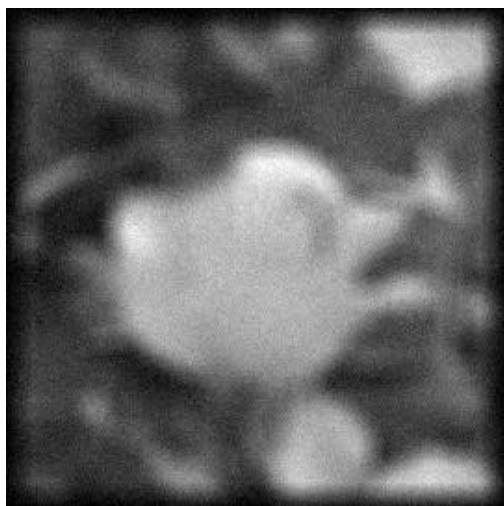


8.4.5 约束最小二乘方滤波

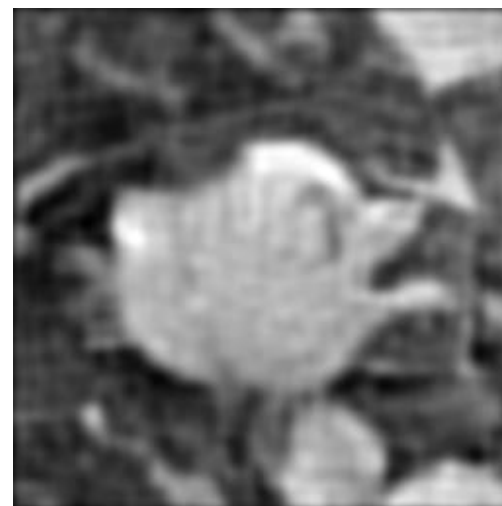
典型图像复原方法

(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(Blurandnoise1,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

8.4.5 约束最小二乘方滤波

典型图像复原方法

(3) 例程

模糊加高斯噪声



约束最小
二乘滤波





8.4.5 约束最小二乘方滤波

典型图像复原方法

(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,[]);



8.4.6 Richardson–Lucy算法

典型图像复原方法

(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



8.4.6 Richardson–Lucy算法

典型图像复原方法

(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)。**



8.4.6 Richardson–Lucy算法

典型图像复原方法

(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');
```



8.4.6 Richardson–Lucy算法

典型图像复原方法

(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');
```

8.4.6 Richardson–Lucy算法

典型图像复原方法

(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



8.5 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。



8.5 盲去卷积复原

(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。



8.5 盲去卷积复原

(2) 原理

■ 代价函数 J :

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

代价函数取最小值对应最优结果



8.5 盲去卷积复原

(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)。
```



8.5 盲去卷积复原

(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');
```

8.5 盲去卷积复原

(3) 例程

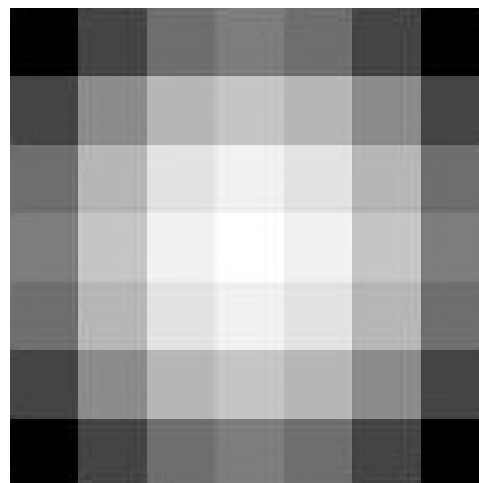
■ 效果



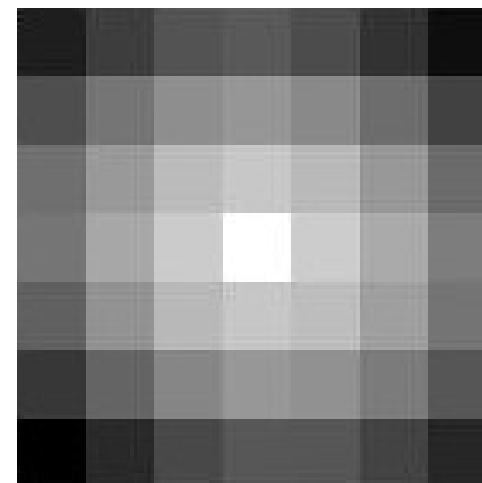
高斯模糊
加噪声



盲复原图像



真正的PSF



重建的PSF



8.6 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



8.6 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

$$\begin{cases} x'_1 = ax_1 + by_1 + c & y'_1 = dx_1 + ey_1 + f \\ x'_2 = ax_2 + by_2 + c & y'_2 = dx_2 + ey_2 + f \\ x'_3 = ax_3 + by_3 + c & y'_3 = dx_3 + ey_3 + f \end{cases}$$



8.6 几何失真校正

(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



8.6 几何失真校正

(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');
```



8.6 几何失真校正

(2) 例程

■ 程序

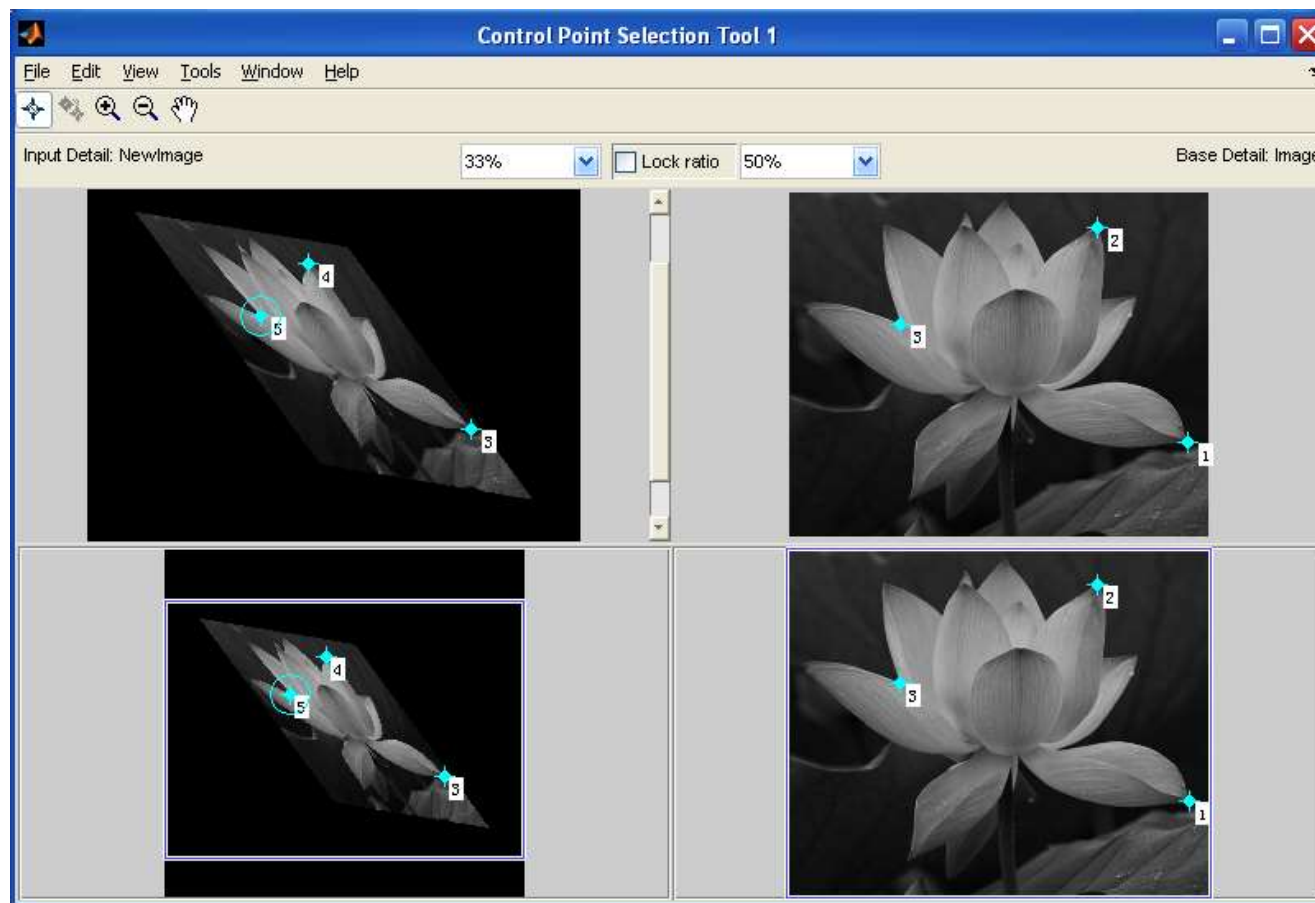
```
cpselect(NewImage,Image);  
input_points=[709 577;409 270;320 370];  
base_points=[487 305;374 41;134 159];  
tform=cp2tform(input_points,base_points,'affine');  
result=imtransform(NewImage,tform,  
                    'XData',[1 w],'YData',[1 h]);  
figure,imshow(result),title('校正后的图像');
```

8.6 几何失真校正



(2) 例程

■ 效果



连接点选择

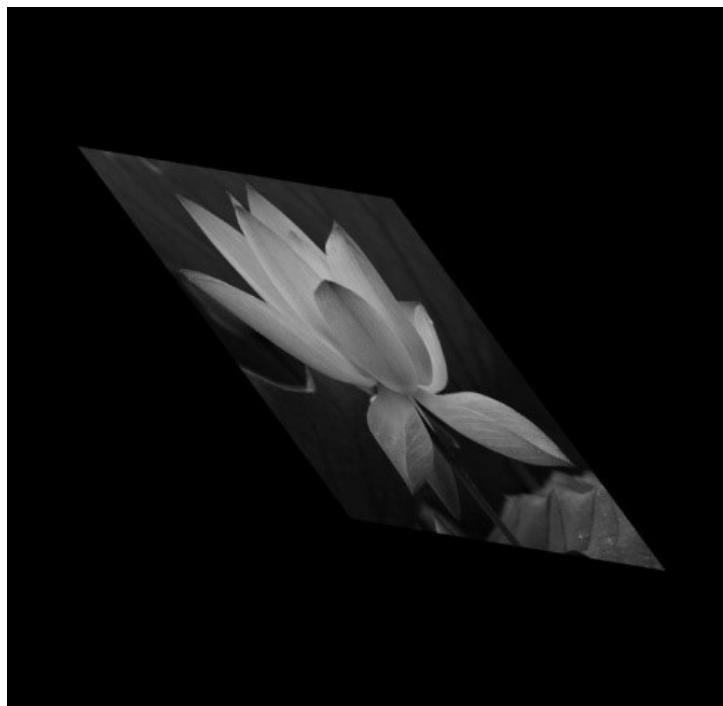
8.6 几何失真校正

(2) 例程

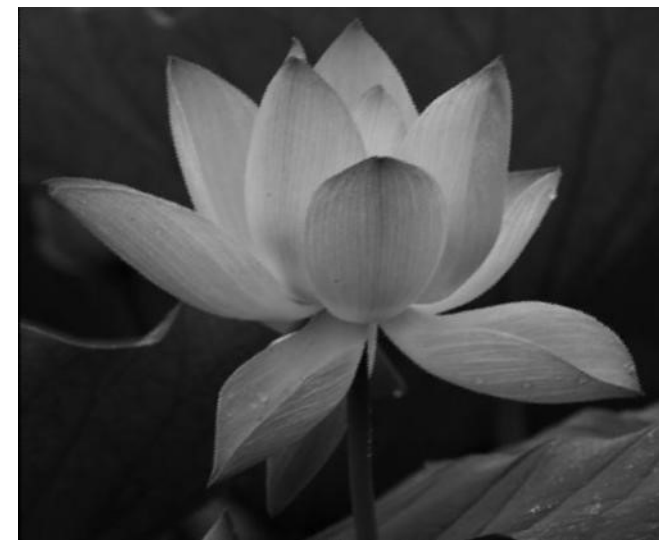
■ 效果



原始图像



几何失真图像



校正后的图像



8.6 几何失真校正

(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。