西安交通大學
XI'AN JIAOTONG UNIVERSITY

数字图像处理

第六次作业

制作成员	班级	学号
赵国梁	自动化 63 班	2160504081

摘要：图像的退化与复原是图像处理的重要分支，处理目标是改善图像，以达到显示或后续处理的某种需求。本文首先讨论图像退化与复原的一般化模型，在此基础上对图像的噪声退化和运动模糊进行具体实现，利用 Matlab 自带的加噪函数分别给图像加入高斯噪声和椒盐噪声。在图像复原方面，主要包括四种均值滤波器，四种统计滤波器和两个自适应滤波器，分别横向和纵向比较这些滤波器的滤波效果，对于逆谐波滤波器还讨论 Q 的取值对滤波效果的影响。接着讨论维纳滤波和约束最小二乘方滤波，然后利用这两种方法对退化图像进行复原，对比两种方法的处理效果以得出各自的优缺点。

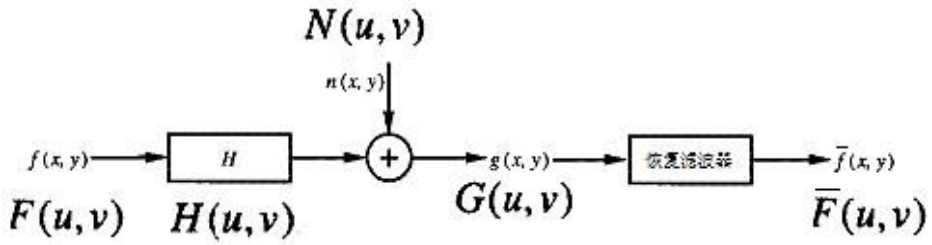
图像恢复技术是图像处理领域一类重要的处理技术，与图像增强等其他基本图像处理技术类似，该技术也是以改善图像视觉质量为目的，所不同的是图像恢复过程需要根据指定的图像退化模型来完成。从该退化模型出发，对在某种情况下退化了的图像进行恢复，以获取未经退化的原始图像。一般地，图像的退化模型可以表述为

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

其中$g(x,y)$表示退化之后的图像，$h(x,y)$表示退化函数，$h(x,y)$表示原始图像，$n(x,y)$表示加性噪声。在空域图像的退化过程是通过与退化函数的卷积和与加性噪声的叠加实现的，若对上式两端同时进行傅里叶变换，那么退化过程的频域表示为

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

在此退化模型的基础上对进行图像回复，只需在输出后添加一个复原滤波器即可，图像的退化与复原过程可以完整地用下图表示。



一．在测试图像上产生高斯噪声 lena 图-需能指定均值和方差；并用多种滤波器恢复图像，分析各自优缺点；

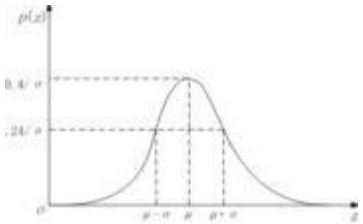
1. 问题分析：

（1）高斯噪声:

高斯噪声是指它的概率密度函数服从高斯分布（即正态分布）的一类噪声。一个高斯随机变量 z 的 PDF 可表示为：

$$P(z)=\frac{1}{\sqrt{2\pi\sigma}}\exp[-\frac{(z-\mu)^2}{2\sigma^2}]$$

其中 z 代表灰度，u 是 z 的均值，是 z 的标准差。高斯噪声的灰度值多集中在均值附近。



（2）算数均值滤波：

令 Sxy 表示中心在点(x,y)处，大小为 m×n 的矩形子图像窗口的一组坐标。算术均值滤波器在 Sxy 定义的区域中计算被污染的图像 g(x,y)的平均值。在点(x,y)处复原图像的值：

$$\hat{f}(x,y)=\frac{1}{mn}\sum_{(s,t)\in S_{xy}}g(s,t)$$

这个操作可以使用大小为 m×n 的一个空间滤波器来实现，其所有系数均为其值的1/mn。均值滤波器平滑一幅图像中的局部变化，虽然模糊了结果，但降低了噪声。

（3）几何均值滤波：

使用几何均值滤波器复原一幅图像由如下表达式给出：

$$\hat{f}(x,y)=\left[\prod_{(s,t)\in S_{xy}} g(s,t)\right]^{\frac{1}{mn}}$$

（4）谐波均值滤波：

谐波均值滤波器操作由如下表达式给出：

$$\hat{f}(x,y)=\frac{mn}{\sum_{(s,t)\in S_{xy}} \frac{1}{g(s,t)}}$$

谐波均值滤波器对于盐粒噪声效果较好，但不适用于胡椒噪声。善于处理像高斯噪声那样的其他噪声。

（5）逆谐波均值滤波：

逆谐波均值滤波器基于如下表达式产生一副复原图像：

$$\hat{f}(x,y)=\frac{\sum_{(s,t)\in S_{xy}} g(s,t)^{Q+1}}{\sum_{(s,t)\in S_{xy}} g(s,t)^{Q}}$$

其中 Q 称为滤波器的阶数。这种滤波器适合减少或者在实际中消除椒盐噪声的影响。当 Q 的值为正时，该滤波器消除胡椒噪声；当 Q 的值为负时，该滤波器消除盐粒噪声。但是它不能同时消除这两种噪声。当 Q 等于 0 时，简化为算数均值滤波器，当 Q 等于-1 时，则为谐波滤波器。

（6）中值滤波：

中值滤波法是一种非线性平滑技术，它将每一像素点的灰度值设置为该点某邻域窗口内的所有像素点灰度值的中值。

中值滤波是基于排序统计理论的一种能有效抑制噪声的非线性信号处理技术，中值滤波的基本原理是把数字图像或数字序列中一点的值用该点的一个邻域中各点值的中值代替，让周围的像素值接近的真实值，从而消除孤立的噪声点。方法是用某种结构的二维滑动模板，将板内像素按照像素值的大小进行排序，生成单调上升（或下降）的为二维数据序列。

$$\hat{f}(x,y)=\underset{(s,t)\in S_{xy}}{median}\{g(s,t)\}$$

（7）最大值和最小值滤波：

选择模板最大的一个像素的滤波器称为最大值滤波器，同理，最小值滤波器是选择模板最小的像素，两个表达式分别如下：

$$\hat{f}(x,y)=\max_{(s,t)\in S_{xy}}\{g(s,t)\}$$

$$\hat{f}(x,y)=\min_{(s,t)\in S_{xy}}\{g(s,t)\}$$

（8）中点滤波：

中点滤波器简单地计算滤波器包围区域中最大值和最小值之间的中点，即

$$\hat{f}(x,y)=\frac{1}{2}\left[\max_{(s,t)\in S_{xy}}\{g(s,t)\}+\min_{(s,t)\in S_{xy}}\{g(s,t)\}\right]$$

（9）修正的阿尔法滤波器：

假设在领域 Sxy 内去掉 $g(s,t)$最低灰度值的 d/2 和最高灰度值的 d/2。令代表剩下的 mn-d 个像素，有这些剩余像素的平均值形成的滤波器称为阿尔法均值滤波器：

$$\hat{f}(\mathrm{x},\mathrm{y})=\frac{1}{mn\text{-}d}\sum_{(\mathrm{s},\mathrm{t})\in S_{xy}}g_r(\mathrm{s},\mathrm{t})$$

其中 d 的范围可取 mn-1。

（10）自适应局部降低噪声滤波器：

自适应局部降低噪声滤波器作用于局部区域 Sxy 。滤波器在该区域中心的任意一点(x ,y)上响应基于以下 4 个量：（a）g(x , y)，带噪图像在点(x ,y)的值；（b）污染 f (x , y) 以形成 g(x , y)的噪声方差；（c）Sxy 中像素的局部均值；（d）Sxy 中像素的局部方差。

如果噪声方差为 0，则滤波器应该简单返回 g(x , y)的值；如果局部方差与是高度相关的，则滤波器返回 g(x , y)的一个近似值；如果两个方差相等，滤波器返回 Sxy 中像素的算术平均值。

$$\hat{f}(\mathrm{x},\mathrm{y})=g(x,y)-\frac{\sigma_\eta^2}{\sigma_L^2}[g(x,y)-m_L]$$

（11）自适应中值滤波器

自适应中值滤波器作用于局部区域 Sxy ，在进行滤波处理时会根据某些条件而改变的

Sxy 尺寸。定义一些变量如下：

$z_{\min}$ = S_{xy}中的最小灰度值

$z_{\max}$ = S_{xy}中的最大灰度值

z_{med} = S_{xy}中灰度值的中值

z_{xy} = 坐标(x, y)的灰度值

$S_{\max}$ = S_{xy}允许的最大尺寸

自适应中值滤波器算法以两个进程工作：

进程 A：

$A_1 = z_{med} - z_{\min}$

$A_2 = z_{med} - z_{\max}$

如果$A_1 > 0$且$A_2 < 0$，则转B进程，否则增大窗口尺寸

如果窗口尺寸$\leq S_{\max}$，则重复A进程，否则输出z_{med}

进程 B：

$B_1 = z_{xy} - z_{\min}$

$B_2 = z_{xy} - z_{\max}$

如果$B_1 > 0$且$B_2 < 0$，则输出z_{xy}，否则输出z_{med}

2. 实验结果：

（1）添加高斯噪声：

原图

加入gaussian噪声后(0 0.01)

加入gaussian噪声后(0 0.05)

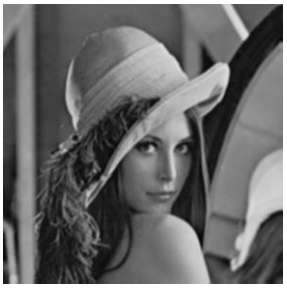
加入gaussian噪声后(0 0.1)

加入gaussian噪声后(0.1 0.01)

加入gaussian噪声后(0.5 0.01)

（2）图像恢复：

算术均值滤波



几何均值滤波

谐波均值滤波

逆谐波均值滤波 Q=1

中值滤波

最大值滤波

最小值滤波

修正阿尔法均值滤波

自适应局部滤波

自适应中值滤波

3. 结果分析：

横向分析：

均值滤波器中，可以看出各个滤波器对高斯噪声的滤波效果差别不大。几何均值滤波器的值小于算术均值滤波器，故图像更黑一些，该滤波器实现的平滑与算术均值滤波器相比，这种处理中丢失的图像细节更少；逆谐波均值滤波器结果偏白，是因为选取的 Q 为正值，相比于谐波均值滤波器，逆谐波均值滤波器处理效果差一些。

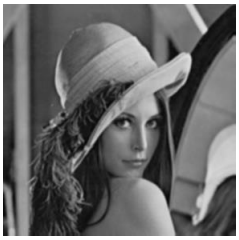
统计排序滤波器中，可以明显看出中值滤波器和修正阿尔法滤波器的效果更好。最大滤波器可以明显看出图像偏白，且一些细节部分加粗，同样，最小滤波器图像偏黑，细节加粗。

自适应滤波器可以明显看出，自适应局部降噪滤波器对高斯噪声的滤波效果要优于自适应中值滤波器。

纵向分析：

算术均值和中值滤波器比较，从下图可以明显看出中值滤波器保留更多图像细节信息，而算数均值在细节方面比较模糊，自适应局部滤波器保留图像的细节最多，但是这样滤波平滑效果受到一定的影响。

算术均值



中值

自适应局部滤波

二．在测试图像 lena 图加入椒盐噪声（椒和盐噪声密度均是 0.1）；用学过的滤波器恢复图像；在使用反谐波分析 Q 大于 0 和小于 0 的作用；

1. 问题分析：

椒盐噪声（salt-and-pepper noise)又称脉冲噪声，它随机改变一些像素值，在二值图像上表现为使一些像素点变白，一些像素点变黑。是由图像传感器，传输信道，解码处理等产生的黑白相间的亮暗点噪声。椒盐噪声往往由图像切割引起，去除脉冲干扰及椒盐噪声最常用的算法是中值滤波。

椒盐噪声的 PDF 由下式给出：

$$p(z)=\begin{cases} p_a, z=a \\ p_b, z=b \\ 1-p_a-p_b, \text{其他} \end{cases}$$

如果 b>a，则灰度级 b 在图像中将显示为一个亮点，反之，灰度级 a 在图像中将显示一个暗点。

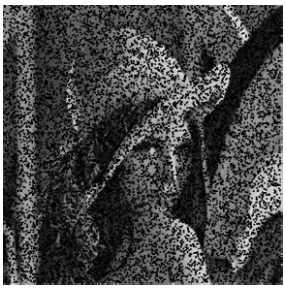
2. 实验结果：

（1）加入椒盐噪声

（2）均值滤波：

算术均值滤波

几何均值滤波



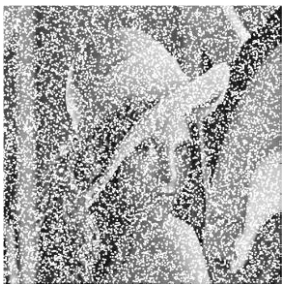
谐波均值滤波

逆谐波均值滤波

（3）统计排序滤波：

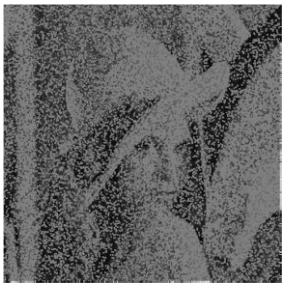
中值滤波

最大值滤波



最小值滤波

中点滤波



（4）反谐波分析 Q 大于 0 和小于 0 的作用

逆谐波**Q**为**1.5**

逆谐波**Q**为**-1.5**

3. 结果分析：

（1）由运行结果可知，四种均值滤波方式均可对椒盐噪声实现衰减，但在恢复效果上存在细微的差别。相比于算术均值与几何均值，尽管对噪声的衰减都起到了作用，但几何均值并未像算术均值那样使图像变得模糊。

（2）由运行结果可知，四种统计滤波方式均可对椒盐噪声实现衰减，但在恢复效果上存在细微的差别。

（3）算术均值与几何均值更适合于处理高斯或均匀随机噪声。逆谐波均值更适合于处理脉冲噪声信号，但它有一个缺点，即必须知道是暗噪声还是亮噪声。

（4）对于逆谐波均值滤波，当 Q 为正时，消除胡椒信号，当 Q 为负时，消除盐粒信号。

（5）中值滤波对椒盐信号有很好的消除作用；最大值滤波消除胡椒信号，最小值滤波消除盐粒信号。

三．推导维纳滤波器并实现下边要求；

(a) 实现模糊滤波器如方程 Eq. (5.6-11).

(b) 模糊 lena 图像：45 度方向，T=1；

(c) 再模糊的 lena 图像中增加高斯噪声（均值=0 方差=10 pixels）以产生模糊图像；

(d)分别利用方程 Eq. (5.8-6)和(5.9-4)，恢复图像；并分析算法的优缺点

1. 问题分析：

（1）模糊滤波器：

图像的运动模糊是由于在成像曝光过程中，物体和相机之间的相对运动，使得物体上同一点的光线在多个成像单元上引起响应造成的。由于相机的成像单元在成像期间完成一个积分过程，以 g(x,y) 表示模糊后的图像，T 表示积分时间，运动模糊原理可以表示为:

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

对上式进行傅里叶变换变换，得

$$G(u,v)=\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\left[\int_0^T f[x-x_0(t),y-y_0(t)]dt\right]e^{-j2\pi(ux+vy)}dxdy$$

利用傅里叶变换的移位性质，频域模糊函数可以表示为

$$H(u,v)=\int_0^t e^{-j2\pi[ux_0(t)+vy_0(t)]}dt$$

现在考虑最简单的情况，那就是如果物体在两个坐标轴方向的运动是匀速直线运动，那么最终的模糊函数可以写为

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

此过程的建模结果与频域滤波形式相类似，在进行处理时可以参照频域滤波的步骤，以 H(x,y)作为滤波器函数，对进行填充和中心化后的图像频谱进行直接相乘即可。

（2）维纳滤波器的推导：

维纳滤波是一种基于最小均方误差准则、对平稳过程的最优估计器。这种滤波器的输出与期望输出之间的均方误差为最小，因此，它是一个最佳滤波系统。它可用于提取被平稳噪声所污染的信号。

图像的退化模型为：

$$x(n_1,n_2)=b(n_1,n_2)*s(n_1,n_2)+w(n_1,n_2) \qquad (1)$$

其中，$s(n_1,n_2)$ 为原始图像，$b(n_1,n_2)$ 为退化函数，$w(n_1,n_2)$ 为噪声函数，$x(n_1,n_2)$ 为退化的图像。并假设 s 与 w 不相关， w 为 0 均值的平稳随机过程。

图像的复原模型为：

$$\hat{s}(n_1,n_2)=h(n_1,n_2)*x(n_1,n_2)=\sum_{l_1}\sum_{l_2}h(l_1,l_2)\times x(n_1-l_1,n_2-l_2) \qquad (2)$$

其中， $\hat{s}(\mathrm{n}_1,\mathrm{n}_2)$ 为恢复的图像， $\mathrm{h}(\mathrm{n}_1,\mathrm{n}_2)$ 为恢复滤波器。

误差度量为：

$$e^2 = E\{(s(n_1,n_2)-\hat{s}(n_1,n_2))^2\} \quad (3)$$

基于正交性原理，若要求误差最小，则必有下式成立：

$$E\{e(n_1,n_2)\times x^*(m_1,m_2)\}=0 \quad (4)$$

将（3）式带入（4）式有：

$$E\{s(n_1,n_2)\times x^*(m_1,m_2)\}=E\{\hat{s}(n_1,n_2)\times x^*(m_1,m_2)\} \quad (5)$$

即

$$\begin{aligned}
&R_{sx}(n_1-m_1,n_2-m_2)=E\{\sum_{l_1}\sum_{l_2}h(l_1,l_2)\times x(n_1-l_1,n_2-l_2)\times x^*(m_1,m_2)\}\\
&=\sum_{l_1}\sum_{l_2}h(l_1,l_2)\times E\{x(n_1-l_1,n_2-l_2)\times x^*(m_1,m_2)\}\\
&=\sum_{l_1}\sum_{l_2}h(l_1,l_2)\times R_{xx}(n_1-l_1-m_1,n_2-l_2-m_2)\\
&=h(n_1-m_1,n_2-m_2)*R_{xx}(n_1-m_1,n_2-m_2)
\end{aligned} \quad (6)$$

换元得：

$$R_{sx}(n_1,n_2)=h(n_1,n_2)*R_{xx}(n_1,n_2) \quad (7)$$

等式两端同时取傅里叶变换得：

$$P_{sx}(w_1,w_2)=H(w_1,w_2)\times P_x(w_1,w_2) \quad (8)$$

即

$$H(w_1,w_2)=\frac{P_{sx}(w_1,w_2)}{P_x(w_1,w_2)} \quad (9)$$

公式（8）中

$$\begin{aligned}
&R_{sx}(n_1,n_2)=E\{s(n_1+k_1,n_2+k_2)\times x^*(k_1,k_2)\}\\
&=E\{s(n_1+k_1,n_2+k_2)\times(\sum_{l_1}\sum_{l_2}b(l_1,l_2)\times s(k_1-l_1,k_2-l_2)+w(k_1,k_2))^*\}\\
&=\sum_{l_1}\sum_{l_2}b^*(l_1,l_2)\times E\{s(n_1+k_1,n_2+k_2)\times s^*(k_1-l_1,k_2-l_2)+s(n_1+k_1,n_2+k_2)\times w^*(k_1,k_2)\}\\
&=\sum_{l_1}\sum_{l_2}b^*(l_1,l_2)\times R_s(n_1+l_1,n_2+l_2)\\
&=b^*(-n_1,-n_2)*R_s(n_1,n_2)
\end{aligned} \quad (10)$$

公式（10）两端同时取傅里叶变换得：

$$P_{sx}(w_1,w_2)=B^*(w_1,w_2)\times P_s(w_1,w_2) \quad (11)$$

公式（8）中

$$
\begin{aligned}
&R_x(n_1,n_2)=E\{x(n_1+k_1,n_2+k_2)\times x^*(k_1,k_2)\}\\
&=E\{(\sum_{l_1}\sum_{l_2}b(l_1,l_2)\times s(n_1+k_1-l_1,n_2+k_2-l_2)+w(n_1+k_1,n_2+k_2))\times(\sum_{m_1}\sum_{m_2}b(m_1,m_2)\times s(k_1-m_1,k_2-m_2)+w(k_1,k_2))^*\}\\
&=\sum_{l_1}\sum_{l_2}\sum_{m_1}\sum_{m_2}b(l_1,l_2)\times b^*(m_1,m_2)\times E\{s(n_1+k_1-l_1,n_2+k_2-l_2)\times s^*(k_1-m_1,k_2-m_2)+R_w(n_1,n_2)\\
&=\sum_{m_1}\sum_{m_2}b^*(m_1,m_2)\sum_{l_1}\sum_{l_2}b(l_1,l_2)R_s(n_1+m_1-l_1,n_2+m_2-l_2)+R_w(n_1,n_2)\\
&=\sum_{m_1}\sum_{m_2}b^*(m_1,m_2)\times(b(n_1+m_1,n_2+m_2)*R_s(n_1+m_1,n_2+m_2))+R_w(n_1,n_2)\\
&=b^*(-n_1,-n_2)*b(n_1,n_2)*R_s(n_1,n_2)+R_w(n_1,n_2)
\end{aligned}
\tag{12}
$$

公式（12）两端同时取傅里叶变换：

$$P_x(w_1,w_2)=|B(w_1,w_2)|^2\times P_s(w_1,w_2)+P_w(w_1,w_2)\tag{13}$$

将（11）式和（13）式带入（8）式得

$$H(w_1,w_2)=\frac{B^*(w_1,w_2)\times P_s(w_1,w_2)}{|B(w_1,w_2)|^2\times P_s(w_1,w_2)+P_w(w_1,w_2)}\tag{14}$$

将符号化成与书中一致的表示

$$
\begin{aligned}
w(u,v)&=\frac{H^*(u,v)\times|F(u,v)|^2}{|H(u,v)|^2\times|N(u,v)|^2+|N(u,v)|^2}\\
&=\frac{H^*(u,v)}{|H(u,v)|^2+\frac{|N(u,v)|^2}{|F(u,v)|^2}}
\end{aligned}
\tag{15}
$$

故表达式由下式给出

$$
\begin{aligned}
\hat{F}(u,v)&=[\frac{H^*(u,v)\times S_f(u,v)}{|H(u,v)|^2\times S_f(u,v)+S_\eta(u,v)}]G(u,v)\\
&=[\frac{H^*(u,v)}{|H(u,v)|^2+S_\eta(u,v)/S_f(u,v)}]G(u,v)\\
&=[\frac{1}{H(u,v)}\frac{|H(u,v)|^2}{|H(u,v)|^2+S_\eta(u,v)/S_f(u,v)}]G(u,v)
\end{aligned}
\tag{16}
$$

（3）约束最小二乘方滤波：

维纳滤波是建立在最小化统计准则的基础上，因此在平均意义上它是最优的，而约束最小二乘方滤波对每一幅图像都能取得最优的效果。一般地，约束最小二乘方滤波器的函数表达式为

$$H_S=\frac{H^*(u,v)}{|H(u,v)|^2+\gamma|P(u,v)|^2}$$

其中 H(u,v)表示图像的退化函数， γ 是一个参数，为取得最佳滤波效果，其调整目标是使得

$$\left\| g - H\hat{f} \right\|^2 = \left\| \eta \right\|^2$$

其中，$\left\| w \right\|^2 \triangleq w^T w$ 是欧几里得向量范数，$\hat{f}$ 是未退化图像的估计。

这个最佳问题在频率域中的解决由下面的表达式给出：

$$\hat{F}(\mathrm{u},\mathrm{v}) = [\frac{H^*(\mathrm{u},\mathrm{v})}{\left|H(\mathrm{u},\mathrm{v})\right|^2 + \gamma \left|P(\mathrm{u},\mathrm{v})\right|^2}]\mathrm{G}(\mathrm{u},\mathrm{v})$$

其中，γ 是一个参数，必须对它进行调整以满足约束条件，$p(u,v)$ 是函数

$$p(\mathrm{x},\mathrm{y}) = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$ 的傅里叶变换。

2. 实验结果：

（1）模糊 lena 图像：45 度方向，T=1；并加入高斯噪声

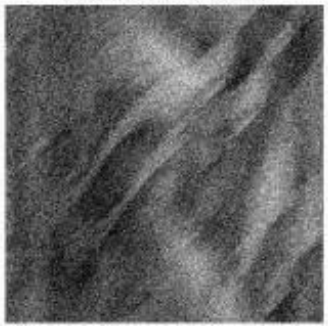
lena.bmp原始图像

运动模糊化**lena.bmp**

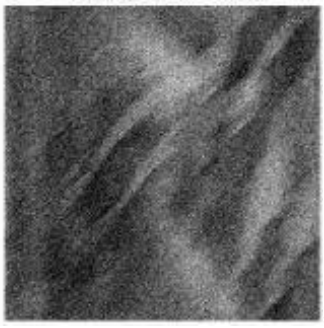
lena.bmp运动模糊+高斯噪声

（2）用维纳滤波恢复原图：

维纳滤波k=0.01



维纳滤波k=0.05



维纳滤波

（3）用约束最小二乘恢复原图：

lena运动模糊+高斯噪声

约束最小二乘滤波

3. 结果分析：

从结果可以直接看出，在相同参数选择下，相比于维纳滤波不同，最小二乘方滤波对噪声的去除效果更好，最终的复原图像中噪声的影响已经非常微弱，这可能与参数选择有关，对于 k 和 r 的选取，过大或者过小都会影响最终滤波的结果。

k=0.005

k=0.001

k=0.0005

r=0.005

r=0.001

r=0.0005

附录：

参考文献：

[1] 冈萨雷斯.数字图像处理（第三版）北京：电子工业出版社，2011
[2] 周品.MATLAB 数字图像处理 北京：清华大学出版社，2012
[3] 杨杰.数字图像处理及 MATLAB 实现 北京：电子工业出版社，2010

代码

1.在测试图像上产生高斯噪声 lena 图-需能指定均值和方差；并用多种滤波器恢复图像，分析各自优缺点；

高斯噪声

```
close all
clc
clear
[I,map]=imread('lena.bmp');
figure(1);
subplot(2,3,1)
imshow(I);
title('原图');
subplot(2,3,2)
I1=imnoise(I,'gaussian',0,0.01);
imshow(I1);
title('加入 gaussian 噪声后(0 0.01)');
I2=imnoise(I,'gaussian',0,0.05);
subplot(2,3,3)
imshow(I2);
title('加入 gaussian 噪声后(0 0.05)');
I3=imnoise(I,'gaussian',0,0.1);
subplot(2,3,4)
imshow(I3);
title('加入 gaussian 噪声后(0 0.1)');
I4=imnoise(I,'gaussian',0.1,0.01);
subplot(2,3,5)
imshow(I4);
title('加入 gaussian 噪声后(0.1 0.01)');
I5=imnoise(I,'gaussian',0.5,0.01);
subplot(2,3,6)
imshow(I5);
title('加入 gaussian 噪声后(0.5 0.01)');
```

高斯噪声恢复原图：

```
lose all; clear all; clc;
I=imread('lena.bmp');
g=imnoise(I,'gaussian',0,0.01);
figure(1)
subplot(1,2,1);imshow(I);title('lena');
subplot(1,2,2);imshow(g);title('lena+gaussian');
[x,y]=size(I);
I=double(I);
A=zeros(x+8,y+8);
A(1,5:y+4)=I(1,:);
A(2,5:y+4)=I(1,:);
A(3,5:y+4)=I(1,:);
A(4,5:y+4)=I(1,:);
A(x+5,5:y+4)=I(x,:);
A(x+6,5:y+4)=I(x,:);
A(x+7,5:y+4)=I(x,:);
A(x+8,5:y+4)=I(x,:);
A(5:x+4,5:y+4)=I;
A(:,1)=A(:,5);
A(:,2)=A(:,5);
A(:,3)=A(:,5);
A(:,4)=A(:,5);
A(:,y+5)=A(:,y+4);
A(:,y+6)=A(:,y+4);
A(:,y+7)=A(:,y+4);
A(:,y+8)=A(:,y+4);
%算术均值滤波器 5*5
A1=zeros(x+8,y+8);
for i=3:x+6
    for j=3:y+6
        B1=A(i-2:i+2,j-2:j+2);
        m1=sum((sum(B1))')/25;
        A1(i,j)=m1;
    end
end
I1=A1(5:x+4,5:y+4);
I1=uint8(I1);
%几何均值滤波器 5*5
A2=zeros(x+8,y+8);
for i=3:x+6
    for j=3:y+6
        B2=A(i-2:i+2,j-2:j+2);
        m2=prod(prod(B2))^0.04;
```

```
            A2(i,j)=m2;
        end
    end
    I2=A2(5:x+4,5:y+4);
    I2=uint8(I2);
%谐波均值滤波器 5*5
    A3=zeros(x+8,y+8);
    for i=3:x+6
        for j=3:y+6
            B3=A(i-2:i+2,j-2:j+2);
            B3=1./B3;
            m3=25/(sum((sum(B3))'));
            A3(i,j)=m3;
        end
    end
    II3=A3(5:x+4,5:y+4);
    I3=255.*II3./max(max(II3)); I3=uint8(I3);
%逆谐波均值滤波器 5*5
    Q=1;
    A4=zeros(x+8,y+8);
    for i=3:x+6
        for j=3:y+6
            B4=A(i-2:i+2,j-2:j+2);
            m4=((sum(sum(B4.^(Q+1)))')/((sum(sum(B4.^Q))');
            A4(i,j)=m4;
        end
    end
    II4=A4(5:x+4,5:y+4);
    I4=255.*II4./max(max(II4));
    I4=uint8(I4);
    figure(2)
    subplot(2,2,1);imshow(I1); title('算术均值滤波器');
    subplot(2,2,2);imshow(I2); title('几何均值滤波器');
    subplot(2,2,3);imshow(I3); title('谐波均值滤波器');
    subplot(2,2,4);imshow(I4); title('逆谐波均值滤波器 Q=1');
%  中值滤波器 5*5
    A5=zeros(x+8,y+8);
    for i=3:x+6
        for j=3:y+6
            B5=A(i-2:i+2,j-2:j+2);
            e=B5(:);
            m5=median(e);
            A5(i,j)=m5;
        end
```

```
    end
    I5=A5(5:x+4,5:y+4);
    I5=uint8(I5);
% 最大值滤波器 5*5
    A6=zeros(x+8,y+8);
    for i=3:x+6
        for j=3:y+6
            B6=A(i-2:i+2,j-2:j+2);
            m6=max(max(B6));
            A6(i,j)=m6;
        end
    end
    I6=A6(5:x+4,5:y+4);
    I6=uint8(I6);
% 最小值滤波器 5*5
    A7=zeros(x+8,y+8);
    for i=3:x+6
        for j=3:y+6
            B7=A(i-2:i+2,j-2:j+2);
            m7=min(min(B7));
            A7(i,j)=m7;
        end
    end
    I7=A7(5:x+4,5:y+4);
    I7=uint8(I7);
% 修正阿尔法滤波器 5*5
    A8=zeros(x+8,y+8);
    for i=3:x+6
        for j=3:y+6
            B8=A(i-2:i+2,j-2:j+2);
            B8=sort(B8(:));
            c=B8(6:20);
            m8=(sum(c))/15;
            A8(i,j)=m8;
        end
    end
    I8=A8(5:x+4,5:y+4);
    I8=uint8(I8);
    figure(3)
    subplot(2,2,1);imshow(I5); title('中值滤波器');
    subplot(2,2,2);imshow(I6); title('最大值滤波器');
    subplot(2,2,3);imshow(I7); title('最小值滤波器');
    subplot(2,2,4);imshow(I8); title('修正阿尔法滤波器');
% 自适应局部降噪滤波器 5*5
```

```
A9=zeros(x+8,y+8);
for i=3:x+6
    for j=3:y+6
        B9=A(i-2:i+2,j-2:j+2);
        m9=mean(B9(:));
        n9=var(B9(:));
        if n9<0.01
            A9(i,j)=A(i,j)-1*(A(i,j)-m9);
        else
            A9(i,j)=A(i,j)-(0.01/n9)*(A(i,j)-m9);
        end
    end
end
II9=A9(5:x+4,5:y+4);
I9=255.*II9./max(max(II9));
I9=uint8(I9);
%自适应中值滤波器
I10=adpmedian(g,7);
I10=uint8(I10);
figure(4)
subplot(1,2,1);imshow(I9); title('自适应局部降噪滤波器 5*5');
subplot(1,2,2);imshow(I10); title('自适应中值滤波器');
```

2.在测试图像 lena 图加入椒盐噪声（椒和盐噪声密度均是 0.1）；

```
close all
clc
clear
[I,map]=imread('lena.bmp');
figure(1);
subplot(1,2,1)
imshow(I);
title('原图');
subplot(1,2,2)
I1=imnoise(I,'salt & pepper',0.1);
imshow(I1);
title('加入椒盐噪声后');
I2=sszz(I1,3);
figure(2)
subplot(2,2,1)
imshow(I2);
title('算术均值滤波');
I3=jhjz(I1,3);
subplot(2,2,2)
imshow(I3);
```

```
title('几何均值滤波');
I4=xbjz(I1,3);
subplot(2,2,3)
imshow(I4);
title('谐波均值滤波');
I5=nxbzz(I1,1.5,3);
subplot(2,2,4)
imshow(I5);
title('逆谐波均值滤波');
figure(3)
subplot(2,2,1)
I6=zzlb(I1,3);
imshow(I6)
title('中值滤波');
I7=max1(I1,3);
subplot(2,2,2)
imshow(I7);
title('最大值滤波');
I8=minlb(I1,3);
subplot(2,2,3)
imshow(I8);
title('最小值滤波');
I9=zdlb(I1,3);
subplot(2,2,4)
imshow(I9);
title('中点滤波');
figure(4)
subplot(1,2,1)
imshow(I5);
title('逆谐波 Q 为 1.5');
I10=nxbzz(I1,-1.5,3);
subplot(1,2,2)
imshow(I10);
title('逆谐波 Q 为-1.5');
```

3.推导维纳滤波器并实现下边要求；

(a) 实现模糊滤波器如方程 Eq. (5.6-11).

(b) 模糊 lena 图像：45 度方向，T=1；

(c) 再模糊的 lena 图像中增加高斯噪声，均值= 0 ，方差=10 pixels 以产生模糊图像；

(d) 分别利用方程 Eq. (5.8-6)和(5.9-4)，恢复图像；并分析算法的优缺点

```
clc;clear;
A=imread('lena.bmp'); f=double(A);
[M,N]=size(A);
P=M;Q=N;
```

```
for x=1:P
      for y=1:Q
            f(x,y)=[(-1)^(x+y)]*f(x,y);
       end
end
F=fft2(f);
T=1;a=0.1;b=0.1;
for u=1:P
      for v=1:Q
  H(u,v)=T/(pi*((u-M/2)*a+(v-N/2)*b))*sin(pi*((u-M/2)*a+(v-N/2)*b))*exp(-i*pi*((u-M/2)*a+(v-N/2)*b));
          if isnan(H(u, v)) == 1
                H(u, v) = 1;
          end
           G(u,v)=H(u,v)*F(u,v);
      end
end
g=real(ifft2(G));
for x=1:P
      for y=1:Q
            g(x,y)=[(-1)^(x+y)]*g(x,y);
      end
end
m=min(min(g));
g=g-m;
g=256*g/max(max(g));
B=uint8(g);
var_n=10*10/512/512;
C=imnoise(B,'gaussian',0,var_n);
figure(1)
subplot(1,3,1);imshow(A);title('lena');
subplot(1,3,2);imshow(B);title('a=0.1 b=0.1 T=1');
subplot(1,3,3);imshow(C);title('gaussian');
```

%wiener

```
g1=double(C);
for x=1:P
      for y=1:Q
            g1(x,y)=[(-1)^(x+y)]*g1(x,y);
      end
end
G1=fft2(g1);
H1 = H .* conj(H);
k=0.001;
```

```
F1 = (G .* H1) ./ (H .* (H1 + k));
f1 = ifft2(F1);
f1 = real(f1);
for x=1:P
    for y=1:Q
        f1(x,y)=[(-1)^(x+y)]*f1(x,y);
    end
end
m1=min(min(f1));
f1=f1-m1;
f1=256*f1/max(max(f1));
D=uint8(f1);
```

%CLSF

```
g2=double(C);
for x=1:P
    for y=1:Q
        g2(x,y)=[(-1)^(x+y)]*g2(x,y);
    end
end
G2=fft2(g2);
p=[0,-1,0;-1,4,-1;0,-1,0];
for x=1:3
    for y=1:3
        p(x,y)=[(-1)^(x+y)]*p(x,y);
    end
end
Pa=fft2(p,512,512);
H1 = H .* conj(H);
P1 = Pa .* conj(Pa);
r=0.001;
F2 = (G .* conj(H)) ./ (H1 + r .* P1); f2 = ifft2(F2);
f2 = real(f2);
for x=1:P
    for y=1:Q
        f2(x,y)=[(-1)^(x+y)]*f2(x,y);
    end
end
m2=min(min(f2));
f2=f2-m2;
f2=256*f2/max(max(f2));
E=uint8(f2);
```

```
figure(2)
subplot(1,2,1);imshow(D);title('wiener k=0.001');
subplot(1,2,2);imshow(E);title('CLSF r=0.001');
```