重庆邮电大学

学生实验实习报告册

学年学期：2022 -2023 学年 □春■秋学期

课程名称：信号处理实验

学生学院：通信与信息工程学院

专业班级：

学生学号：

学生姓名：

联系电话：

重庆邮电大学教务处制

课程名称	信号处理实验	课程编号	A2010550
实验地点	YF314	实验时间	2022 年 10 月 21 日
校外指导教师	郑丹玲	校内指导教师	郑丹玲
实验名称	系统响应及系统稳定性		
评阅人签字		成绩	

一、实验目的

（1）学会运用 Matlab 求解离散时间系统的零状态响应；

（2）学会运用 Matlab 求解离散时间系统的单位取样响应；

（3）学会运用 Matlab 求解离散时间系统的卷积和。

二、实验原理

1.离散时间系统的响应

离散时间 LTI 系统可用线性常系数差分方程来描述，即

$$\sum_{i=0}^{N} a_i y(n-i) = \sum_{j=0}^{N} b_j x(n-j)$$

其中，$a_i(i=0,1,...,N)$和$b_i(i=0,1,...,N)$为实常数。

Matlab 中函数 filter 可对（2-1）的差分方程，对在指定时间范围内的输入序列所产生的响应进行求解。函数 filter 的语句格式为

```
y=filter(b,a,x)
```

其中，x 为输入的离散序列；y 为输出的离散序列；y 的长度与 x 的长度一样；b 与 a 分别为差分方程右端与左端的系数向量。

2. 离散时间系统的单位取样响应

系统的单位取样响应定义为系统在$\delta(n)$激励下系统的零状态响应，用 h(n)表示。Matlab 求解单位取样响应可利用函数 filter，并将激励设为单位抽样序列。

Matlab 另一种求单位取样响应的方法是利用控制系统工具箱提供的函数 impz 来实现。impz 函数的常用语句格式为

```
impz(b,a,N)
```

其中，参数 N 通常为正整数，代表计算单位取样响应的样值个数。

3. 离散时间信号的卷积和运算

由于系统的零状态响应是激励与系统的单位取样响应的卷积，因此卷积运算在离散时间信号处理领

域被广泛应用。离散时间信号的卷积定义为

$$y(n)=x(n)*h(n)=\sum_{m=-\infty}^{\infty}x(m)h(n-m)$$

可见，离散时间信号的卷积运算是求和运算，因而常称为“卷积和”。Matlab 求离散时间信号卷积和的命令为 conv，其语句格式为

y=conv(x,h)

其中，x 与 h 表示离散时间信号值的向量；y 为卷积结果。用 Matlab 进行卷积和运算时，无法实现无限的累加，只能计算时限信号的卷积。

对于给定函数的卷积和，应计算卷积结果的起始点及其长度。两个时限序列的卷积和长度等于两个序列长度的和减 1。

除此之外，还存在着二维卷积，Matlab 中的二维卷积函数 conv2()

（1）C=conv2(A,B)

设矩阵 A 的大小为 Ma×Na，矩阵 B 的大小为卷积 Mb×Nb，则卷积后 C 的大小为（Ma+Mb-1）×（Na+Nb-1）；

（2）C=conv2(Hcol,Hrow,A)

矩阵 A 先与 Hcol 向量在列方向进行卷积，然后再与 Hrow 向量在行方向上进行卷积；

（3）C=conv2(A,B,'shape')

当 shape=full 时，返回全部二维卷积结果，即返回 C 的大小为（Ma+Mb-1）×（Na+Nb-1）当 shape=same 时，返回与 A 同样大小的卷积中心部分；当 shape=valid 时，不考虑边界补零，即只要有边界补出的零参与运算的都舍去，返回 C 的大小为（Ma+Mb-1）×（Na+Nb-1）conv2 函数常用于对图像进行滤波处理。

三、实验程序

```
clc;
clear;

a = [3 4 1];
b = [1 1];
figure(1);
subplot(2,1,1);
impz(b,a,30);
grid on;
title("系统取样响应");

c = [2.5 6 10];
```

```
d = [1];
subplot(2,1,2);
impz(d,c,30);
grid on;
title("系统取样响应");

nh = -3:14;
nx = -3:9;
h = ((7/8).^nh).*(uDT(nh)-uDT(nh-10));
x = uDT(nx)-uDT(nx-5);
% 求卷积
y = conv(x,h);
ny1 = nx(1) + nh(1);
ny2 = nx(end) + nh(end);
ny = ny1:ny2;

figure(2);
subplot(3,1,1);
stem(nh,h,'fill');
grid on;
xlabel("k");
ylabel("h(k)");
title("h(k)");

subplot(3,1,2);
stem(nx,x,'fill');
grid on;
xlabel("k");
ylabel("x(k)");
title("x(k)");

subplot(3,1,3);
stem(ny,y,'fill');
grid on;
xlabel("k");
ylabel("x(k)*h(k)");
title("x(k)*h(k)");


%% 图像
lena = imread('lena_colour.bmp');
figure(3);
subplot(2,2,1);
imshow(lena);
title("原始图像");
Gx = [-1 0 1;-2 0 2;-1 0 1];
```

```
Gy = [1 2 1;0 0 0;-1 -2 -1];

lena_gx(:,:,1) = conv2(Gx,lena(:,:,1));
lena_gx(:,:,2) = conv2(Gx,lena(:,:,2));
lena_gx(:,:,3) = conv2(Gx,lena(:,:,3));
subplot(2,2,2);
imshow(lena_gx);
title("采用 Gx 进行卷积滤波后的图像");


lena_gy(:,:,1) = conv2(Gy,lena(:,:,1));
lena_gy(:,:,2) = conv2(Gy,lena(:,:,2));
lena_gy(:,:,3) = conv2(Gy,lena(:,:,3));
subplot(2,2,3);
imshow(lena_gy);
title("采用 Gy 进行卷积滤波后的图像");

lena_gxy(:,:,1) = (lena_gy(:,:,1) + lena_gx(:,:,1))/2;
lena_gxy(:,:,2) = (lena_gy(:,:,2) + lena_gx(:,:,2))/2;
lena_gxy(:,:,3) = (lena_gy(:,:,3) + lena_gx(:,:,3))/2;

% lena_gxy(:,:,1) = sqrt(lena_gy(:,:,1).* lena_gy(:,:,1) + lena_gx(:,:,1).
      * lena_gx(:,:,1));
% % lena_gxy(:,:,2) = sqrt(lena_gy(:,:,2).* lena_gy(:,:,2) + lena_gx(:,:,2
      ).* lena_gx(:,:,2));
% lena_gxy(:,:,3) = sqrt(lena_gy(:,:,3).* lena_gy(:,:,3) + lena_gx(:,:,3).
      * lena_gx(:,:,3));


subplot(2,2,4);
imshow(lena_gxy);
title("采用 Gx 和 Gy 进行卷积滤波后的图像");

clc;
clear;
x = [3,11,7,0,-1,4,2];
nx = -3:3;
h = [2 3 0 -5 2 1];
nh = -1:4;
subplot(3,1,1);
stem(nh,h,'fill');
grid on;
xlabel("k");
ylabel("h(k)");
title("系统序列 h(k)");
subplot(3,1,2);
```

```
stem(nx,x,'fill');
grid on;
xlabel("k");
ylabel("x(k)");
title("输入序列 x(k)");
y = conv(x,h);
nn = nx(1) + nh(1):nx(end) + nh(end);

subplot(3,1,3);
stem(nn,y,'fill');
grid on;
xlabel("k");
ylabel("x(k)*h(k)");
title("零状态响应序列 x(k)*h(k)");
% [y ny] = Convolution(x,h,nx,nh);

%% 图像其余滤波方式

lena = imread('lena_colour.bmp');
J=imnoise(lena,'salt & pepper',0.02);
figure(2);
subplot(3,2,1);
imshow(lena);
title("原始图像");
subplot(3,2,2),
imshow(J);
title('添加盐椒噪声')

%% 中值滤波
b(:,:,1) = medfilt2(J(:,:,1),[3,3]);
b(:,:,2) = medfilt2(J(:,:,2),[3,3]);
b(:,:,3) = medfilt2(J(:,:,3),[3,3]);
subplot(3,2,3),
imshow(b);
title('中值滤波');
%% 拉氏算子锐化图像
h=[0,1,0;1,-4,0;0,1,0];%拉氏算子
c(:,:,1) = filter2(h,lena(:,:,1));
c(:,:,2) = filter2(h,lena(:,:,2));
c(:,:,3) = filter2(h,lena(:,:,3));
subplot(3,2,4),
imshow(c);
title('拉氏算子');

%% prewitt 算子
H = fspecial('prewitt');%应用 prewitt 算子锐化图像
```

```
d(:,:,1) = filter2(H,lena(:,:,1));
d(:,:,2) = filter2(H,lena(:,:,2));
d(:,:,3) = filter2(H,lena(:,:,3));
subplot(3,2,5),
imshow(d);
title('prewitt 算子');


%% log 算子
H=fspecial('log');%应用 log 算子锐化图像
e(:,:,1) = filter2(H,lena(:,:,1));
e(:,:,2) = filter2(H,lena(:,:,2));
e(:,:,3) = filter2(H,lena(:,:,3));
subplot(3,2,6),
imshow(e);
title('log 算子');
```

四、仿真分析

（1）试用 Matlab 命令求解以下离散时间系统的单位取样响应，并判断系统的稳定性。

1）$3y(n)+4y(n-1)+y(n-2)=x(n)+x(n-1)$

2）$\frac{5}{2}y(n)+6y(n-1)+10y(n-2)=x(n)$

上述系统的系统函数分别为 $\frac{z}{3z+1}$、$\frac{2z^2}{5z^2+12z+20}$，可以发现系统 1）的极点分别位于单位圆内，系统 2）的极点均位于单位圆外，所以最终从系统函数的极点可以反映出系统 1）是稳定的，系统 2）是不稳定的。

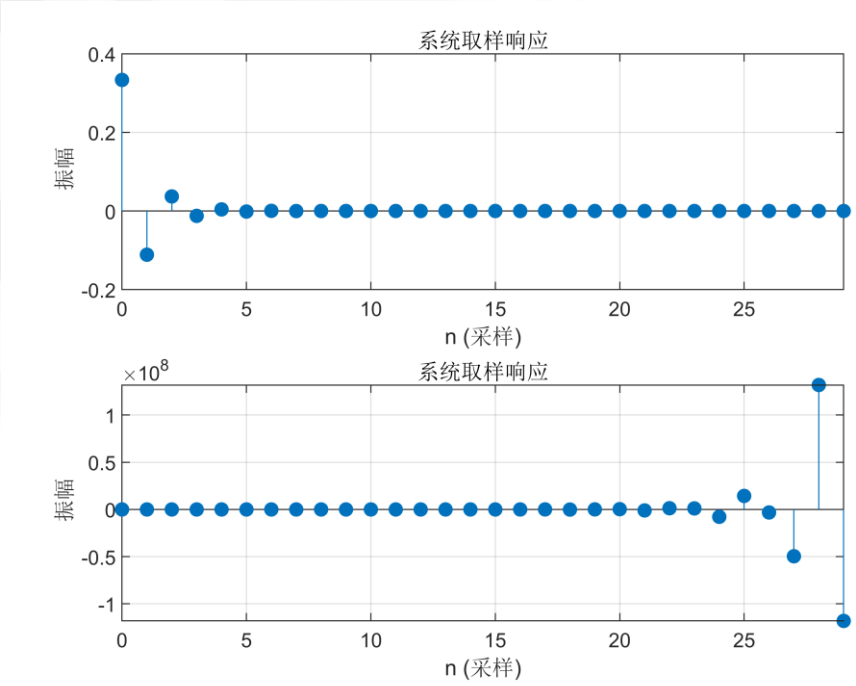

图 1 离散时间系统的单位取样响应仿真结果图

由图 1 可以发现，系统 1）取样响应随着取样点的不断增加而趋近于 0，而系统 2）取样响应则随着取样点的增加震荡程度不断增加，从仿真结果中可以得出系统 1）是稳定的，系统 2）是不稳定的，与理论分析结果相同。

（2）已知某系统的单位取样响应为$h(n)=\left(\frac{7}{8}\right)^n[u(n)-u(n-10)]$，试 MATLAB 求当激励信号为

$x(n)=u(n)-u(n-5)$时，求系统的零状态响应。

系统的零状态响应可以使用卷积进行求解，即$y(n)=x(n)*h(n)=\sum_{m=-\infty}^{\infty}x(m)h(n-m)$

此时，将题中输入与系统函数代入卷积中，最终求得

$$y(n)=8\left[1-\frac{7}{8}^{n+1}\right]u(n)-8\left[\frac{7}{8}^{10}-\frac{7}{8}^{n+1}\right]u(n-10)+8\left[1-\frac{7}{8}^{n-4}\right]u(n-5)-8\left[\frac{7}{8}^{10}-\frac{7}{8}^{n-4}\right]u(n-15)$$

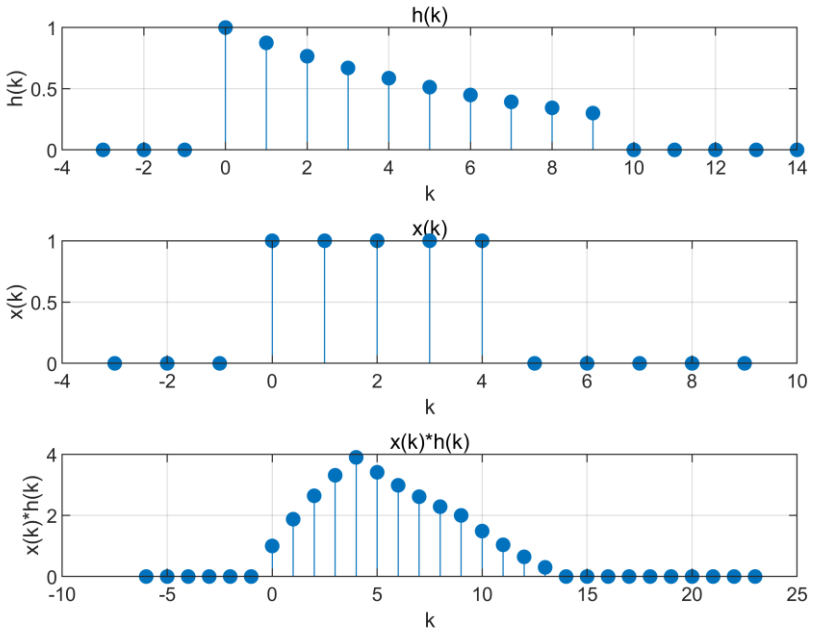

图 2 零状态响应仿真结果图

将上述求解结果与仿真结果图进行对比，发现仿真结果与表达式结果相同，即理论求解等于仿真数值求解，即实验结果精确性高。

（3)用数字图像处理中的 Sobel 算子，调用 conv2 函数对 lena 图像进行滤波。设两个 3×3 的 Sobe 矩阵算子 Gx=[-1 0 1;-2 0 2;-1 0 1]，Gy=[1 2 1;0 0 0;-1 -2 -1]。

1）读取 lena 图像数据，保存到 B 矩阵中；

2）分如下三种情况对 B 进行卷积滤波

① 采用 Gx 与 B 进行卷积；

② 采用 Gy 与的 B 进行卷积；

③ 先采用 Gx 与 B 进行卷积，然后再采用 Gy 与 B 进行卷积；

④ 在一个 figure 中，分四个子图显示出：原始图像，经过 Gx 卷积滤波后图像、经过 Gy 卷积滤波后图像，先后采用 Gx 和 Gy 与的 B 进行卷积滤波后的图像，观察滤波后的图像的效果。

Sobel 算子是图像检测的重要算子之一，其本质是梯度运算。

水平方向：将图像 I 与奇数大小的模板进行卷积，结果为 G_x，比如，模板大小为 3 时，有

$$G_x=\begin{bmatrix}-1 & 0 & +1\\ -2 & 0 & +2\\ -1 & 0 & +1\end{bmatrix}*I$$

垂直方向：将图像 I 与奇数大小的模板进行卷积，结果为 G_y，比如，模板大小为 3 时，有

$$G_y=\begin{bmatrix}+1 & +2 & +1\\ 0 & 0 & 0\\ -1 & -2 & -1\end{bmatrix}*I$$

那么在图像的每一点处，结合以上两个结果可以求出：

$$G=\sqrt{G_x^{\,2}+G_y^{\,2}}$$

经过上述步骤后，最终得到仿真如下图 3 所示。

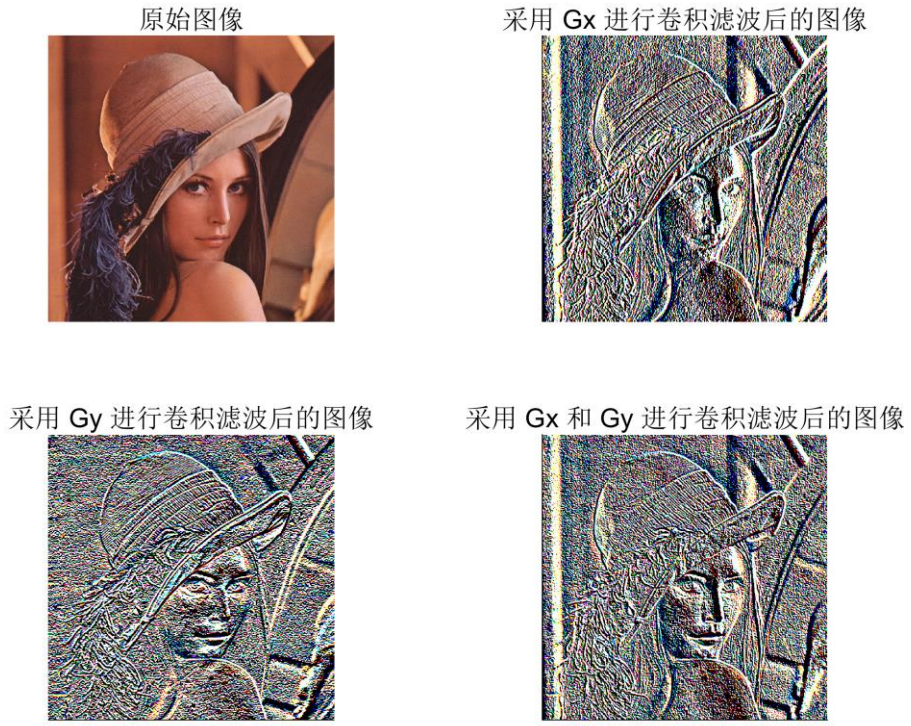

图 3 Sobel 算子滤波仿真结果图

从图 3 可以发现分别利用 sobel 算子分别对 lena 图像进行 x 轴、y 轴以及整体进行边缘检测，可以发现最终利用 Gx 与 Gy 卷积整体边缘检测的效果最佳。

五、思考题

（1） Matlab 的工具箱函数 conv，能用于计算两个有限长序列之间的卷积，但 conv 函数假定这两个序列都从 n=0 开始。试编写 M 文件计算 $x(n)=[3,11,7,0,-1,4,2],-3\leqslant n\leqslant 3$ 和 $h(n)=[2,3,0,-5,2,1],-1\leqslant n\leqslant 4$ 之间的卷积，并绘制 $y(n)$ 的波形图。

3 11 7 0 -1 4 2

2 3 0 -5 2 1

---

3 11 7 0 -1 4 2

6 22 14 0 -2 8 4

-15 -55 -35 0 5 -20 -10

9 33 21 0 -3 12 6

6 22 14 0 -2 8 4

---

6 31 47 6 -51 -5 51 21 -22 -3 8 2

利用**不进位乘法**，可以得到最终 y(n)序列为：

$$y(n)=[6,31,47,6,-51,-5,51,21,-22,-3,\ 8,2],-4\leqslant n\leqslant 7$$

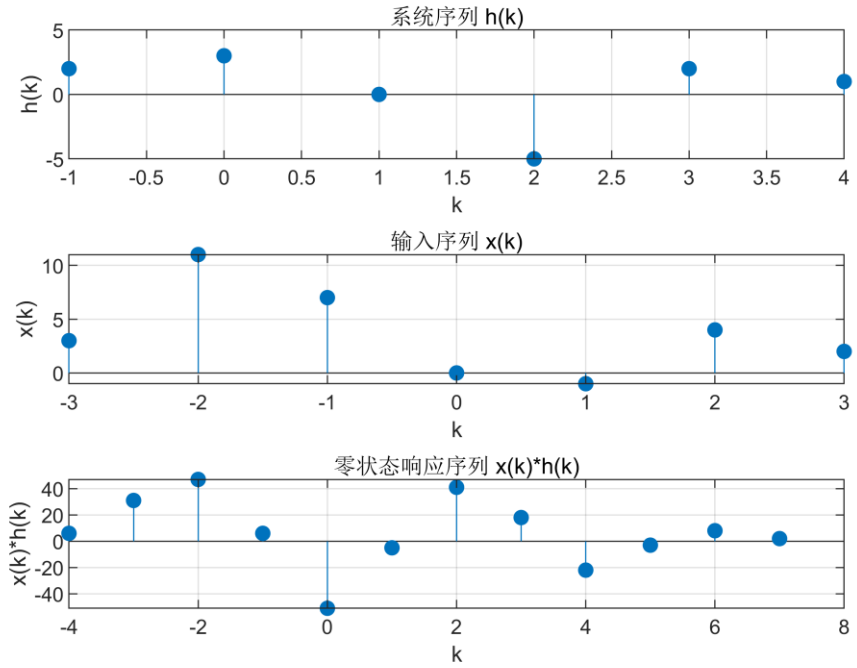

图 4 序列卷积仿真结果图

最终可以明显看出理论结果等于仿真结果。

（2） 在数字图像处理边缘检测技术中，除了 Sobel 算子外，还有哪些边缘检测算子？通过查找资料，选择某种边缘检测算子，在 Matlab 平台上编程实现对 lena 或其他图像进行边缘检测，显示出原图和边缘检测后的图像。

1）中值滤波

中值滤波会选取数字图像或数字序列中像素点及其周围临近像素点（一共有奇数个像素点）的像素值，将这些像素值排序，然后将位于中间位置的像素值作为当前像素点的像素值，让周围的像素值接近真实值，从而消除孤立的噪声点。

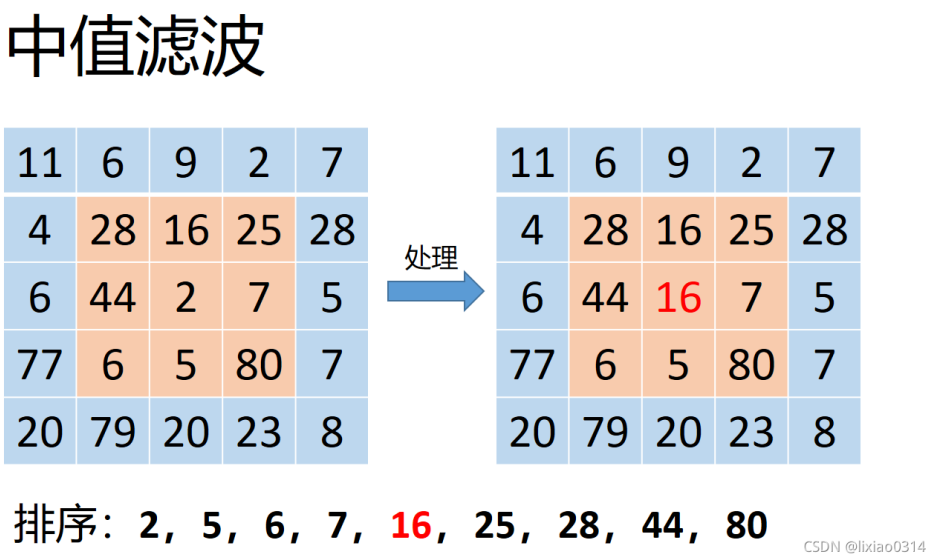

图 5 中值滤波示意图

2）拉式算子滤波

着重于图像中的灰度突变区域，而非灰度级缓慢变化的区域，会产生暗色背景中叠加有浅辉边界线和突变点（轮廓）。原图加拉普拉斯算子计算后的图像可以使图像锐化。

0	1	0
1	-4	1
0	1	0

图 6 拉式算子滤波模板图

3）其他算子滤波

同理，其他算子滤波形式如下图所示。

$$dx=\begin{Bmatrix}1 & 0 & -1\\ 1 & 0 & -1\\ 1 & 0 & -1\end{Bmatrix} \quad dy=\begin{Bmatrix}-1 & -1 & -1\\ 0 & 0 & 0\\ 1 & 1 & 1\end{Bmatrix}$$

图 7 Prewitt 算子滤波模板图

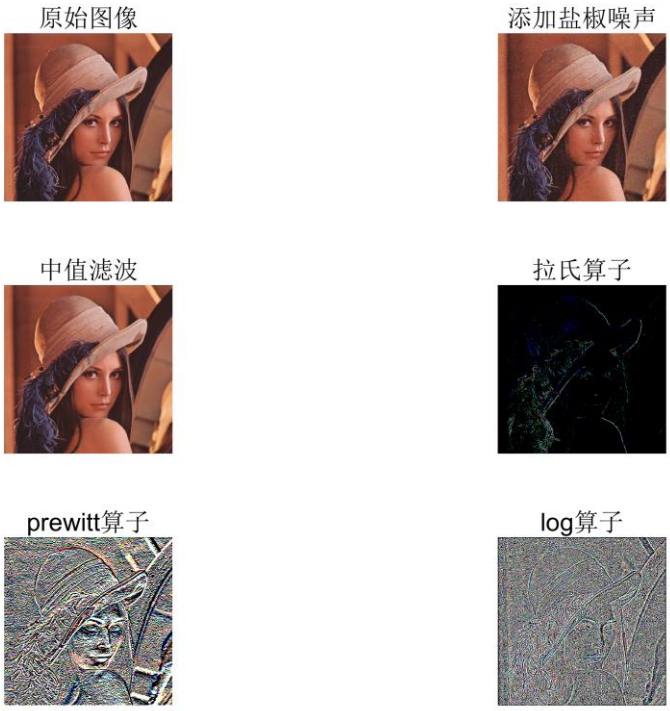

图 7 其他滤波方式仿真结果图

由图 7 可以看出中值滤波可以很好的出去原有含有图像校验噪声的情况下将图像还原，而拉式算子、prewitt 算子、log 算子分别以不同的方式对图像进行了边缘检测，得到不同的边缘检测效果。