|  |  |  |
| --- | --- | --- |
| **得分** | **教师签名** | **批改日期** |
|  |  |  |

课程编号： IB00078

****

**深圳技术大学实验报告**

**课程名称： 数字图像处理**

**实验名称： 图像复原实验**

**班 级： 计算机科学与技术2班**

**指导教师： 李蒙**

**报 告 人： 黄荣权 学号： 202002020213**

**合 作 者： 组号：**

**实验地点： C5-357**

**实验时间： 2022 年 05 月 03 日 星期 二**

**提交时间： 2022/05/03**

|  |
| --- |
| 1. **实验目的：**   1. 了解图像降质/复原处理的模型。  2. 了解估计降质函数的基本原理。  3. 掌握降质图像中常见噪声模型及参数估计方法、基本原理、实现步骤。  4. 加深对几种常用的图像复原方法的理解。  **二、实验原理：**  1. 图像复原的一般过程为：  **分析退化原因**  **建立退化模型**  **反向推演**  **恢复图像**  2. 复原方法  对于线性移不变系统，在空域中图像降质过程通常建模为卷积形式，若已知点扩散函数和加性噪声，就很容易反解出原图像函数。由于在空域中直接复原存在大规模计算问题，所以复原一般通过对图像进行傅里叶变换后，在频域中进行图像复原。   1. 无约束复原——维纳滤波(最小均方误差)   维纳滤波也称为最小二乘滤波，它是使原始图像与其恢复图像之间的均方误差最小的复原方法。对图像进行维纳滤波主要是为了消除图像中存在的噪声。  (2) 约束复原——平滑约束复原(约束最小平方滤波)  约束最小平方滤波是一种比较容易实现的线性复原方法，约束复原除要求了解关于降质系统的传递函数之外，还需要知道某些噪声的统计特性或噪声与图像的某些相关情况。   1. 自适应中值滤波   **三、实验内容步骤**（记录实验主要步骤，且在调试成功后，将结果截屏或拍照保存）   1. 读入选择图像库中一幅灰度图像，对图像用“motion”（运动模糊）及“disk”（ 散焦模糊）类型进行模糊降质，显示模糊前后的图像。   lena = imread('lena.jpg');  lena\_gray=rgb2gray(lena);  subplot(1,3,1);  imshow(lena\_gray);  title('原始图像');  PSF = fspecial('motion',15,45);  h1 = fspecial('disk',40);  J1 = imfilter(lena\_gray,PSF,'circular');  J2 = imfilter(lena\_gray,h1,'circular');  subplot(1,3,2);  imshow(J1);  title('模糊运动图像');  subplot(1,3,3);  imshow(J2);  title('disk模糊图像');   1. 生成大小为的棋盘格式图像（自行查阅checkerboard函数的使用方法），对该图像进行模糊加噪，分别用点扩散函数、NSR、NCORR和ICORR为参数对降质图像进行恢复，显示并对比恢复结果。   %生成一个120×120的棋盘图像  chessboard=checkerboard(10,60,60);  PSF1=fspecial('motion',7,45);  GB1=imfilter(chessboard,PSF1,'circular');  subplot(2,3,1);  imshow(chessboard);  title('棋盘格式图像');  %对棋盘图像进行加噪  noise\_mean=0;  noise\_var=0.05;  GB1=imnoise(GB1,'gaussian',noise\_mean,noise\_var);  subplot(2,3,2);  imshow(GB1);  title('模糊加噪图像');  % 使用点扩散函数恢复降质图像  frest1=deconvwnr(GB1,PSF1);  subplot(2,3,3);  imshow(frest1);  title('点扩散函数恢复降质图像');  Sn=abs(fft2(GB1)).^2;  nA=sum(Sn(:))/numel(GB1); %噪声平均功率1  Sf=abs(fft2(chessboard)).^2; %图像功率谱1  fA=sum(Sf(:))/numel(chessboard); %图像平均功率1  R=nA/fA;%计算噪声和信号比例  % 使用NSR恢复降质图像  fr2=deconvwnr(GB1,PSF,R);%使用deconvwnr函数生成常数比率的维纳滤波图像  subplot(2,3,4);  imshow(fr2);  title('NSR恢复降质图像');  % 使用NCORR和ICORR恢复降质图像  NCORR=fftshift(real(ifft2(Sn)));  ICORR=fftshift(real(ifft2(Sf)));  fr3=deconvwnr(GB1,PSF,NCORR,ICORR);  subplot(2,3,5);  imshow(fr3);  title('NCORR和ICORR恢复降质图像');     1. 对内容1中降质图像采用维纳滤波（deconvwnr）和最小二乘方滤波（deconvreg）的方法复原，显示复原后的图像。   lena = imread('lena.jpg');  lena\_gray=rgb2gray(lena);  subplot(2,2,1);  imshow(lena\_gray);  title('原始图像');  LEN1=12;  THREN1=22;  PFS2=fspecial("motion",LEN1,THREN1);  fuzzy\_lena=imfilter(lena\_gray,PFS2,'circular'); %对图像进行模糊劣化  subplot(2,2,2);  imshow(fuzzy\_lena);  title('模糊劣化图像');  deconvwnr\_lena=deconvwnr(fuzzy\_lena,PFS2);  subplot(2,2,3);  imshow(deconvwnr\_lena);  title('采用维纳滤波复原图像');  least\_squares\_lena=deconvreg(fuzzy\_lena,PFS2);  subplot(2,2,4);  imshow(least\_squares\_lena);  title('采用最小二乘法复原图像')     1. 编写自适应中值滤波adpmedfilt(g, Smax)，分析自适应中值滤波的优点。   中值滤波器受滤波窗口大小影响较大，用于消除噪声和保护图像细节，两者会存在冲突，而自适应中值滤波能根据预设好的条件，动态地改变中值滤波器的窗口尺寸，以同时兼顾去噪声作用和保护细节的效果。  %自适应中值滤波函数  function f = adpmedian (g, Smax)  if (Smax <= 1) || (Smax/2 == round(Smax/2)) || (Smax ~= round(Smax))  error ('SMAX必须是一个大于1的奇数整数.')  end  f = g;  f(:) = 0;  alreadyProcessed = false (size(g));  for k = 3:2:Smax  zmin = ordfilt2(g, 1, ones(k, k),'symmetric');  zmax = ordfilt2(g, k \* k, ones(k, k), 'symmetric');  zmed = medfilt2(g, [k k], 'symmetric');  processUsingLevelB = (zmed > zmin) & (zmax > zmed) &...  ~alreadyProcessed;  zB = (g > zmin) & (zmax > g);  outputZxy = processUsingLevelB & zB;  outputZmed = processUsingLevelB & ~zB;  f (outputZxy) = g(outputZxy);  f (outputZmed) = zmed(outputZmed);  alreadyProcessed = alreadyProcessed | processUsingLevelB;  if all (alreadyProcessed (:))  break;  end  end  f (~alreadyProcessed) = zmed (~alreadyProcessed);  end  lena = imread('lena.jpg');  lena\_gray=rgb2gray(lena);  subplot(1,3,1);  imshow(lena\_gray);  title('原始图像');  LEN1=50;  THREN1=100;  PFS2=fspecial("motion",LEN1,THREN1);  fuzzy\_lena=imfilter(lena\_gray,PFS2,'circular'); %对图像进行模糊劣化  median\_filtering\_lena=adpmedian(fuzzy\_lena,15);  subplot(1,3,2);  imshow(fuzzy\_lena);  title('劣化图像');  subplot(1,3,3);  imshow(median\_filtering\_lena);  title('中值滤波图像')     1. 从图像库中读取三幅灰度图像，对每幅依次添加椒盐噪声、高斯噪声、均匀分布噪声，观察图像的变化。然后依次用均值滤波，中值滤波、自适应中值滤波和基于局部区域统计特征的自适应滤波方法（wiener2）对噪声图像进行处理，并比较处理后的结果。通过比较总结出自适应滤波的优势和适用的滤波场合。   **自适应滤波的优势:不但能够滤除概率较大的椒盐噪声，而且能够更好的保**  **图像的细节，这是常规的中值滤波器做不到的。**  **适用的滤波场合: 在图像噪声密度不是很大的情况下。**  %读取原始图像  wallpaper1=imread('wallpaper1.jpg');  %将图像转换为灰度图像  wallpaper1=rgb2gray(wallpaper1);  % 分别向wallpaperr1添加椒盐噪音、高斯噪声、均匀分布噪声  wallpaper1\_salt\_pepper=imnoise(wallpaper1,'salt & pepper',0.05);  wallpaper1\_gauwallpaperian=imnoise(wallpaper1,'gaussian',0.05);  wallpaper1\_speckle=imnoise(wallpaper1,'speckle',0.05);  % 对椒盐噪声图像依次用均值滤波，中值滤波、自适应中值滤波和二维自适应去噪滤波进行处理  wallpaper1\_salt\_pepper\_mean\_filtering=filter2(y11,wallpaper1\_salt\_pepper)/255;  wallpaper1\_salt\_pepper\_median\_filtering=medfilt2(wallpaper1\_salt\_pepper,[3,3]);  wallpaper1\_salt\_pepper\_adaptive\_median\_filtering=adpmedian(wallpaper1\_salt\_pepper,15);  wallpaper1\_salt\_pepper\_wiener2\_filtering = wiener2(wallpaper1\_salt\_pepper,[5 5]);  % 对高斯噪声图像依次用均值滤波，中值滤波、自适应中值滤波和二维自适应去噪滤波进行处理  wallpaper1\_gauwallpaperian\_mean\_filtering=filter2(y11,wallpaper1\_salt\_pepper)/255;  wallpaper1\_gauwallpaperian\_median\_filtering=medfilt2(wallpaper1\_salt\_pepper,[3,3]);  wallpaper1\_gauwallpaperian\_adaptive\_median\_filtering=adpmedian(wallpaper1\_salt\_pepper,15);  wallpaper1\_gauwallpaperian\_wiener2\_filtering = wiener2(wallpaper1\_salt\_pepper,[5 5]);  % 对均匀分布噪声图像依次用均值滤波，中值滤波、自适应中值滤波和二维自适应去噪滤波进行处理  wallpaper1\_speckle\_mean\_filtering=filter2(y11,wallpaper1\_salt\_pepper)/255;  wallpaper1\_speckle\_median\_filtering=medfilt2(wallpaper1\_salt\_pepper,[3,3]);  wallpaper1\_speckle\_adaptive\_median\_filtering=adpmedian(wallpaper1\_salt\_pepper,15);  wallpaper1\_speckle\_wiener2\_filtering = wiener2(wallpaper1\_salt\_pepper,[5 5]);  subplot(5,3,1);imshow(wallpaper1\_salt\_pepper);title('添加椒盐噪音');  subplot(5,3,2);imshow(wallpaper1\_gauwallpaperian);title('添加高斯噪声');  subplot(5,3,3);imshow(wallpaper1\_speckle);title('添加均匀分布噪音');  subplot(5,3,4);imshow(wallpaper1\_salt\_pepper\_mean\_filtering);title('均值滤波');  subplot(5,3,5);imshow(wallpaper1\_speckle\_mean\_filtering);title('均值滤波');  subplot(5,3,6);imshow(wallpaper1\_salt\_pepper\_wiener2\_filtering);title('均值滤波');  subplot(5,3,7);imshow(wallpaper1\_salt\_pepper\_mean\_filtering);title('中值滤波');  subplot(5,3,8);imshow(wallpaper1\_speckle\_mean\_filtering);title('中值滤波');  subplot(5,3,9);imshow(wallpaper1\_salt\_pepper\_wiener2\_filtering);title('中值滤波');  subplot(5,3,10);imshow(wallpaper1\_salt\_pepper\_mean\_filtering);title('自适应中值滤波');  subplot(5,3,11);imshow(wallpaper1\_speckle\_mean\_filtering);title('自适应中值滤波');  subplot(5,3,12);imshow(wallpaper1\_salt\_pepper\_wiener2\_filtering);title('自适应中值滤波');  subplot(5,3,13);imshow(wallpaper1\_salt\_pepper\_mean\_filtering);title('二维自适应去噪滤波');  subplot(5,3,14);imshow(wallpaper1\_speckle\_mean\_filtering);title('二维自适应去噪滤波');  subplot(5,3,15);imshow(wallpaper1\_salt\_pepper\_wiener2\_filtering);title('二维自适应去噪滤波');    %读取原始图像  wallpaper2=imread('wallpaper2.jpg');  %将图像转换为灰度图像  wallpaper2=rgb2gray(wallpaper2);  % 分别向wallpaperr1添加椒盐噪音、高斯噪声、均匀分布噪声  wallpaper2\_salt\_pepper=imnoise(wallpaper2,'salt & pepper',0.05);  wallpaper2\_gauwallpaperian=imnoise(wallpaper2,'gaussian',0.05);  wallpaper2\_speckle=imnoise(wallpaper2,'speckle',0.05);  % 对椒盐噪声图像依次用均值滤波，中值滤波、自适应中值滤波和二维自适应去噪滤波进行处理  wallpaper2\_salt\_pepper\_mean\_filtering=filter2(y11,wallpaper2\_salt\_pepper)/255;  wallpaper2\_salt\_pepper\_median\_filtering=medfilt2(wallpaper2\_salt\_pepper,[3,3]);  wallpaper2\_salt\_pepper\_adaptive\_median\_filtering=adpmedian(wallpaper2\_salt\_pepper,15);  wallpaper2\_salt\_pepper\_wiener2\_filtering = wiener2(wallpaper2\_salt\_pepper,[5 5]);  % 对高斯噪声图像依次用均值滤波，中值滤波、自适应中值滤波和二维自适应去噪滤波进行处理  wallpaper2\_gauwallpaperian\_mean\_filtering=filter2(y11,wallpaper2\_salt\_pepper)/255;  wallpaper2\_gauwallpaperian\_median\_filtering=medfilt2(wallpaper2\_salt\_pepper,[3,3]);  wallpaper2\_gauwallpaperian\_adaptive\_median\_filtering=adpmedian(wallpaper2\_salt\_pepper,15);  wallpaper2\_gauwallpaperian\_wiener2\_filtering = wiener2(wallpaper2\_salt\_pepper,[5 5]);  % 对均匀分布噪声图像依次用均值滤波，中值滤波、自适应中值滤波和二维自适应去噪滤波进行处理  wallpaper2\_speckle\_mean\_filtering=filter2(y11,wallpaper2\_salt\_pepper)/255;  wallpaper2\_speckle\_median\_filtering=medfilt2(wallpaper2\_salt\_pepper,[3,3]);  wallpaper2\_speckle\_adaptive\_median\_filtering=adpmedian(wallpaper2\_salt\_pepper,15);  wallpaper2\_speckle\_wiener2\_filtering = wiener2(wallpaper2\_salt\_pepper,[5 5]);  subplot(5,3,1);imshow(wallpaper2\_salt\_pepper);title('添加椒盐噪音');  subplot(5,3,2);imshow(wallpaper2\_gauwallpaperian);title('添加高斯噪声');  subplot(5,3,3);imshow(wallpaper2\_speckle);title('添加均匀分布噪音');  subplot(5,3,4);imshow(wallpaper2\_salt\_pepper\_mean\_filtering);title('均值滤波');  subplot(5,3,5);imshow(wallpaper2\_speckle\_mean\_filtering);title('均值滤波');  subplot(5,3,6);imshow(wallpaper2\_salt\_pepper\_wiener2\_filtering);title('均值滤波');  subplot(5,3,7);imshow(wallpaper2\_salt\_pepper\_mean\_filtering);title('中值滤波');  subplot(5,3,8);imshow(wallpaper2\_speckle\_mean\_filtering);title('中值滤波');  subplot(5,3,9);imshow(wallpaper2\_salt\_pepper\_wiener2\_filtering);title('中值滤波');  subplot(5,3,10);imshow(wallpaper2\_salt\_pepper\_mean\_filtering);title('自适应中值滤波');  subplot(5,3,11);imshow(wallpaper2\_speckle\_mean\_filtering);title('自适应中值滤波');  subplot(5,3,12);imshow(wallpaper2\_salt\_pepper\_wiener2\_filtering);title('自适应中值滤波');  subplot(5,3,13);imshow(wallpaper2\_salt\_pepper\_mean\_filtering);title('二维自适应去噪滤波');  subplot(5,3,14);imshow(wallpaper2\_speckle\_mean\_filtering);title('二维自适应去噪滤波');  subplot(5,3,15);imshow(wallpaper2\_salt\_pepper\_wiener2\_filtering);title('二维自适应去噪滤波');    %读取原始图像  wallpaper3=imread('wallpaper3.jpg');  %将图像转换为灰度图像  wallpaper3=rgb2gray(wallpaper3);  % 分别向wallpaperr1添加椒盐噪音、高斯噪声、均匀分布噪声  wallpaper3\_salt\_pepper=imnoise(wallpaper3,'salt & pepper',0.05);  wallpaper3\_gauwallpaperian=imnoise(wallpaper3,'gaussian',0.05);  wallpaper3\_speckle=imnoise(wallpaper3,'speckle',0.05);  % 对椒盐噪声图像依次用均值滤波，中值滤波、自适应中值滤波和二维自适应去噪滤波进行处理  wallpaper3\_salt\_pepper\_mean\_filtering=filter2(y11,wallpaper3\_salt\_pepper)/255;  wallpaper3\_salt\_pepper\_median\_filtering=medfilt2(wallpaper3\_salt\_pepper,[3,3]);  wallpaper3\_salt\_pepper\_adaptive\_median\_filtering=adpmedian(wallpaper3\_salt\_pepper,15);  wallpaper3\_salt\_pepper\_wiener2\_filtering = wiener2(wallpaper3\_salt\_pepper,[5 5]);  % 对高斯噪声图像依次用均值滤波，中值滤波、自适应中值滤波和二维自适应去噪滤波进行处理  wallpaper3\_gauwallpaperian\_mean\_filtering=filter2(y11,wallpaper3\_salt\_pepper)/255;  wallpaper3\_gauwallpaperian\_median\_filtering=medfilt2(wallpaper3\_salt\_pepper,[3,3]);  wallpaper3\_gauwallpaperian\_adaptive\_median\_filtering=adpmedian(wallpaper3\_salt\_pepper,15);  wallpaper3\_gauwallpaperian\_wiener2\_filtering = wiener2(wallpaper3\_salt\_pepper,[5 5]);  % 对均匀分布噪声图像依次用均值滤波，中值滤波、自适应中值滤波和二维自适应去噪滤波进行处理  wallpaper3\_speckle\_mean\_filtering=filter2(y11,wallpaper3\_salt\_pepper)/255;  wallpaper3\_speckle\_median\_filtering=medfilt2(wallpaper3\_salt\_pepper,[3,3]);  wallpaper3\_speckle\_adaptive\_median\_filtering=adpmedian(wallpaper3\_salt\_pepper,15);  wallpaper3\_speckle\_wiener2\_filtering = wiener2(wallpaper3\_salt\_pepper,[5 5]);  subplot(5,3,1);imshow(wallpaper3\_salt\_pepper);title('添加椒盐噪音');  subplot(5,3,2);imshow(wallpaper3\_gauwallpaperian);title('添加高斯噪声');  subplot(5,3,3);imshow(wallpaper3\_speckle);title('添加均匀分布噪音');  subplot(5,3,4);imshow(wallpaper3\_salt\_pepper\_mean\_filtering);title('均值滤波');  subplot(5,3,5);imshow(wallpaper3\_speckle\_mean\_filtering);title('均值滤波');  subplot(5,3,6);imshow(wallpaper3\_salt\_pepper\_wiener2\_filtering);title('均值滤波');  subplot(5,3,7);imshow(wallpaper3\_salt\_pepper\_mean\_filtering);title('中值滤波');  subplot(5,3,8);imshow(wallpaper3\_speckle\_mean\_filtering);title('中值滤波');  subplot(5,3,9);imshow(wallpaper3\_salt\_pepper\_wiener2\_filtering);title('中值滤波');  subplot(5,3,10);imshow(wallpaper3\_salt\_pepper\_mean\_filtering);title('自适应中值滤波');  subplot(5,3,11);imshow(wallpaper3\_speckle\_mean\_filtering);title('自适应中值滤波');  subplot(5,3,12);imshow(wallpaper3\_salt\_pepper\_wiener2\_filtering);title('自适应中值滤波');  subplot(5,3,13);imshow(wallpaper3\_salt\_pepper\_mean\_filtering);title('二维自适应去噪滤波');  subplot(5,3,14);imshow(wallpaper3\_speckle\_mean\_filtering);title('二维自适应去噪滤波');  subplot(5,3,15);imshow(wallpaper3\_salt\_pepper\_wiener2\_filtering);title('二维自适应去噪滤波'); |
| **四、实验总结与心得体会**  通过这次实验，我了解图像降质和复原处理的模型以及估计降质函数的基本原理，掌握降质图像中常见噪声模型及参数估计方法、基本原理、实现步骤。对多张图像添加不同噪声之后进行修复，使我加深了图像复原方法的理解。 |