中国地质大学（武汉）

数字图像处理实验报告

学　　院：自动化学院

课　　程：数字图像处理

指导老师：上官星辰

学　　号：20201000128

班　　级：231202

姓　　名：刘瑾瑾

2023 年 11 月 20 日

目录

一、实验内容

1、数字图像的读取与保存；
2、图像彩色空间的转换；
3、图像几何变换与插值；
4、图像的平滑与锐化。

二、实验目的

1、掌握数字图像处理的基础知识，图像的色彩空间与转换，图像的空间几何变换，傅里叶、 离散余弦和小波变换，正确使用图像质量的评价指标；
2、基本掌握图像增强方法，了解图像退化模型及模型参数估计，掌握图像复原方法；

三、实验内容

3.1 任务一

3.1.1 实验要求

读入一幅 RGB 图像，变换为灰度图像和二值图像，并在同一个窗口内分成三个子窗口来分别显示 RGB 图像和灰度图像，注上文字标题。

3.1.2 实验结果

图 1 任务一结果

实验结果如图 1 所示，从左至右依次为原始图像、灰度图像和二值化图像。原始图像为彩色图像，即 RGB 图像，具有红色、绿色和蓝色三个色彩通道，可以转换变为灰度图。二值图只具有 0 或 1，是在灰度图基础上设置阈值转换而来，上图二值图效果良好。

3.2 任务二

3.2.1 实验要求

读入一幅灰度图像，对其进行傅里叶变换和一级小波分解变换，并在同一个窗口内分成三个子窗口来分别显示原始灰度图像、傅里叶变换和一级小波分解变换的图像，注上文字标题。

3.2.2 实验结果

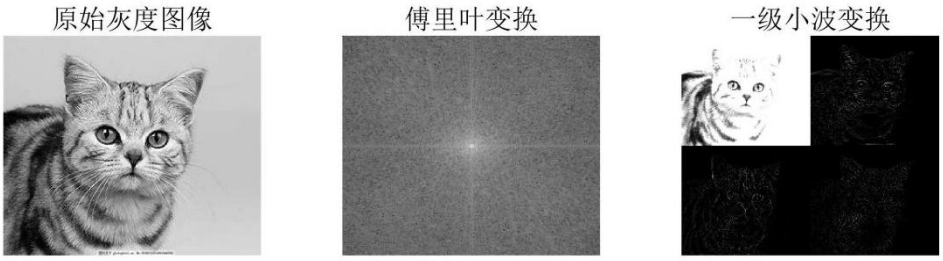

图 2 任务二结果

实验结果如图 2 所示，从左至右依次为原始灰度图像、傅里叶变换图像和一级小波变换图像。其中小波变换结果包括近似分量、水平分量、垂直分量和对角分量四个分量，由于图较小显示为黑色，故单独放大显示在图 3 中。

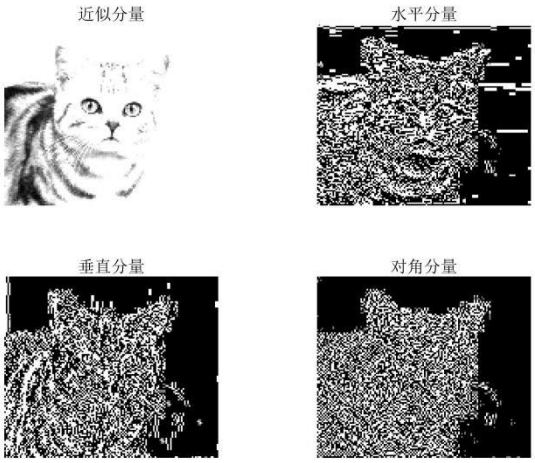

图 3 小波变换结果

3.3 任务三

3.3.1 实验要求

读入一幅 RGB 图像，编写 MATLAB 程序实现下列变换： 图像向上、向右平移，并用白色填充空白部分，再对其做水平镜像，然后旋转 45 度， 再缩小 4 倍。最后，在同一个窗口内分成两个子窗口来分别显示原图像和经历复合变换后的图像，注上文字标题。

3.3.2 实验结果

图 4 任务三结果

实验结果如图 4 所示，图像先向上和向右移动 50 个像素单位，后用白色填充空白部分，再做水平镜像，旋转 45°，最后缩小 4 倍，与实验要求一致。

3.4 任务四

3.4.1 实验要求

读入一幅 RGB 图像，编写 MATLAB 程序实现下列要求：加入高斯噪声和乘性噪声，输出图像；选择合适的方法分别对高斯和乘性噪声去噪，然后输出去噪图像。

3.4.2 实验结果

高斯噪声和乘性噪声是数字图像处理中常见的两种噪声类型，它们对图像质量产生不同的影响。高斯噪声是一种统计性噪声，其分布服从高斯分布，通常由于电子元件的随机震动、温度变化、光照变化等因素引起，在图像中呈现为像素

值的随机扰动，会使图像中的像素值呈现随机的涨落，使图像看起来更加模糊。乘性噪声是一种与图像原始像素值相关的噪声，可能由于图像采集设备的非线性特性、传感器故障等因素引起，它会使图像中的每个像素值乘以一个随机值，会导致图像的对比度降低，使图像细节丧失。

实验结果如图 5 所示，从左至右依次为原始图像、加入高斯噪声图像和加入乘性噪声图像。

图 5 加入噪声后图像

对加入高斯噪声和乘性噪声的图像分别进行均值滤波、高斯滤波、中值滤波、维纳滤波和小波变换滤波，结果分别为图 6 和图 7。

图 6 高斯噪声滤波结果

图 7 乘性噪声滤波结果

如图 6 所示，小波变换滤波相对于中值滤波、高斯滤波、中值滤波、维纳滤波的效果更好。由于小波变换的多尺度分析特性，可以更好地处理不同频率上的噪声，对于去除特定尺度上的噪声效果较好。

如图 7，维纳滤波效果仅次于小波变换滤波，原因在于维纳滤波在已知噪声模型的情况下，可以更好地进行去噪。对于不同的噪声类型，适用不同的滤波方法，但是选择合适的参数也至关重要。在实际应用中，应根据噪声特点选择合适的滤波方法和合适的参数。

3.5 任务五

3.5.1 实验要求

读入一幅 RGB 图像，编写 MATLAB 程序实现下列要求：采用三种不同算子对图像进行锐化处理，在同一窗口输出图像；使用同态滤波器实现图像增强，输出图像。

3.5.2 实验结果

分别选择 Sobel 算子、Prewitt 算子和 Laplacian 算子对原始图像进行锐化操作，

对灰度图和彩色图的处理结果分别如图 8 和图 9 所示，可以看出 Sobel 算子锐化效果最好，Prewitt 算子次之，Laplacian 算子较模糊，彩色图像具有彩色边缘。

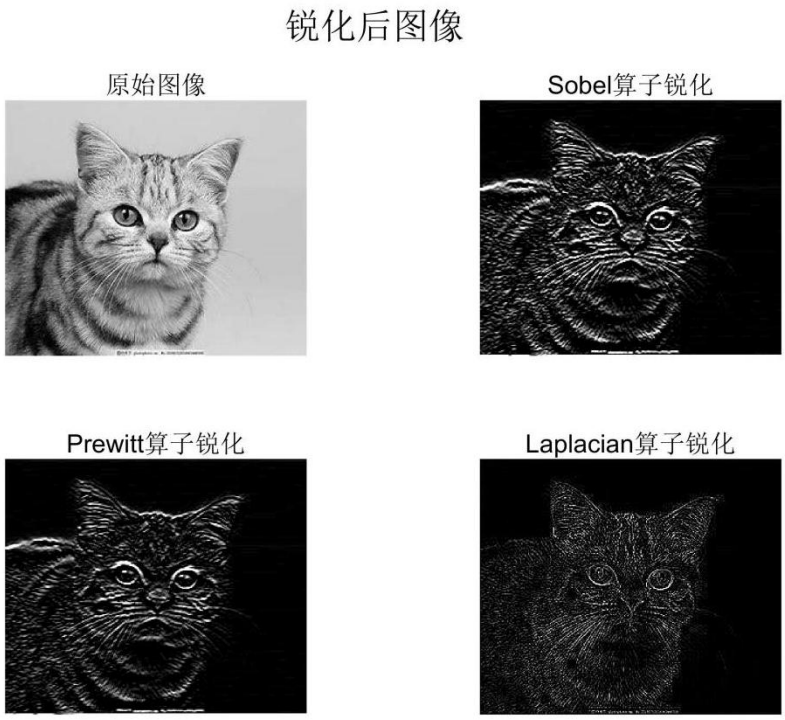

图 8 灰度图锐化后结果

图 9 彩色图锐化后结果

将锐化后图像和原始图像叠加，如图 10 和 11 所示：

图 10 灰度图锐化图像与原图像叠加

图 11 彩色图锐化图像与原始图像叠加

同态滤波的核心思想是利用图像的对数变换将图像分解为光照（低频）和反射（高频）两个部分，通过对这两个部分进行滤波来达到目的。同态滤波的过程如下：（1）对数变换： 对原始图像进行对数变换，将其转换到对数域。对数变换能够将光照和反射分离开。（2）频域滤波： 在对数域中，通过应用滤波器进行滤波操作。通常使用高通滤波器来增强反射分量，使用低通滤波器来平滑光照分量。（3）反对数变换： 对滤波后的结果进行反对数变换，将其转换回原始图像域。按照同态滤波的过程编写代码，对灰度图和彩色图片进行同态滤波，运行结果分别如图 12 和图 13 所示：

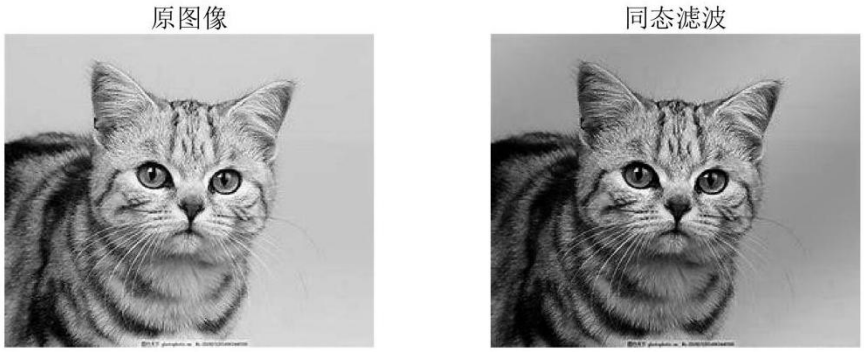

图 12 灰度图同态滤波结果

图 13 彩色图同态滤波结果

彩色图像使用同态滤波时，会生成三张输出图像，对应于原始图像中的三个通道（红色、绿色、蓝色），如图 13 中间图像所示。这三张图像可以合并为一张彩色图像，得到最终的同态滤波结果，即为图 13 右侧所示。这个过程保持了图像的彩色信息，并对每个通道的光照和细节进行独立的调整，从而提高图像的对比度。

四、实验代码

```
%% 任务一
%读入一幅 RGB 图像，变换为灰度图像和二值图像，并在同一个窗口内分成三个
%子窗口来分别显示 RGB 图像和灰度图像，注上文字标题。
figure(1)
img= imread('cat2.jpg');
subplot(1,3,1)
imshow(img)
title('原始图像')
%转换为灰度图
gray = rgb2gray(img);
subplot(1,3,2)
imshow(gray)
title('灰度图像')
imwrite(gray,'gray.jpg')%保存灰度图
%二值化图像
BW = imbinarize(gray,'adaptive'); %自适应阈值二值化
subplot(1,3,3)
imshow(BW)
title('二值化图像')
% 展示整个图像窗口
%sgtitle('任务一');

%% 任务二
%读入一幅灰度图像，对其进行傅里叶变换和一级小波分解变换，并在同一个窗口内分成三
个
%子窗口来分别显示原始灰度图像、傅里叶变换和一级小波分解变换的图像，注上文字标题

figure(2)
gray_img=imread('gray.jpg');
subplot(1, 3, 1);
imshow(gray_img);
title('原始灰度图像');
% 进行傅里叶变换
```

```
fft_img = fft2(double(gray_img));%计算傅里叶变换
fft_img = fftshift(fft_img);%对结果进行移位，以便在中心显示低频分量
magnitude=log(1+abs(fft_img));
subplot(1, 3, 2);
imshow(magnitude,[]);
title('傅里叶变换');
% 进行一级小波分解
waveletLevel = 1;
[LL, LH, HL, HH] = dwt2(gray_img,'db1'); % 以'db1'为例，你可以根据需要选择其他小波基
LL=uint8(LL);
subplot(1, 3, 3);
imshow([LL, LH; HL, HH]);
title('一级小波变换');

figure(21)
subplot(2,2,1)
imshow(LL);
title('近似分量');

subplot(2,2,2)
imshow(LH);
title('水平分量');

subplot(2,2,3)
imshow(HL);
title('垂直分量');

subplot(2,2,4)
imshow(HH);
title('对角分量');
% 展示整个图像窗口
%sgtitle('任务二');

%% 任务三
```

```
%读入一幅 RGB 图像，编写 MATLAB 程序实现下列变换：图像向上、向右平移，并用白色填充
%空白部分，再对其做水平镜像，然后旋转 45 度，再缩小 4 倍。最后，在同一个窗口内分成
%两个子窗口来分别显示原图像和经历复合变换后的图像，注上文字标题。

img=imread('cat.jpg');
%图像向上平移
shiftedUp = imtranslate(img, [0, -50], 'FillValues', 255);
%图像向右平移
shiftedRight = imtranslate(shiftedUp, [50, 0], 'FillValues', 255);
%图像水平镜像
mirroredImage = flip(shiftedRight, 2);%水平翻转
%旋转 45 度
rotatedImage = imrotate(mirroredImage, 45);%旋转 45°
%缩小四倍
scaledImage = imresize(rotatedImage, 0.25);
figure(3);
% 显示原 RGB 图像
subplot(1, 2, 1);
imshow(img);
title('原图像');
% 显示经历复合变换后的图像
subplot(1, 2, 2);
imshow(scaledImage);
title('复合变换图像');
% 展示整个图像窗口
%sgtitle('任务三');

%% 任务三
%读入一幅 RGB 图像，编写 MATLAB 程序实现下列要求：加入高斯噪声和乘性噪声，输出
%图像；选择合适的方法分别对高斯和乘性噪声去噪，然后输出去噪图像。
% 读入 RGB 图像
rgbImage = imread('cat.jpg');
% 显示原图像
```

```
figure(4);
subplot(1, 3, 1);
imshow(rgbImage);
title('原始图像');
% 加入高斯噪声
gaussianNoise = imnoise(rgbImage, 'gaussian', 0, 0.02);
subplot(1, 3, 2);
imshow(gaussianNoise);
title('高斯噪声图像');
% 加入乘性噪声
multiplicativeNoise = imnoise(rgbImage, 'speckle',0.02);
subplot(1, 3, 3);
imshow(multiplicativeNoise);
title('乘性噪声图像');

gaussianNoise=multiplicativeNoise;
% 去除高斯噪声
figure(5);
%均值滤波
filteredImage1 = imfilter(gaussianNoise, fspecial('average', [3
3]));
% 高斯滤波
sigma = 1;
filteredImage2 = imgaussfilt(gaussianNoise, sigma);
% 中值滤波
for i = 1:3 % 遍历每个颜色通道
    filteredImage3(:, :, i) = medfilt2(gaussianNoise(:, :, i), [3 3]);
end
%维纳滤波
for i = 1:3 % 遍历每个颜色通道
    filteredImage4(:, :, i) = wiener2(gaussianNoise(:, :, i), [3 3]);
end
%小波滤波
% 将图像转换为双精度类型
rgbImageDouble = im2double(rgbImage);
```

```
% 小波变换
[LL, LH, HL, HH] = dwt2(rgbImageDouble, 'haar'); % 使用 Haar 小波

% 对 LL 子带进行滤波操作，这里使用简单的高斯滤波作为示例
filterSize = 5;
filter = fspecial('gaussian', filterSize, 2);
LLFiltered = imfilter(LL, filter, 'same', 'conv');

% 重构滤波后的 RGB 图像
filteredImage5 = idwt2(LLFiltered, LH, HL, HH, 'haar');

subplot(2,3,1);
imshow(gaussianNoise);
%title('高斯噪声图像')
title('乘法噪声图像')

subplot(2,3,2);
imshow(filteredImage1);
title('均值滤波')

subplot(2,3,3);
imshow(filteredImage2);
title('高斯滤波')

subplot(2,3,4);
imshow(filteredImage3);
title('中值滤波')

subplot(2,3,5);
imshow(filteredImage4);
title('维纳滤波')

subplot(2,3,6);
imshow(filteredImage5);
```

```
title('小波变换滤波')

%% 任务五
%读入一幅 RGB 图像，编写 MATLAB 程序实现下列要求： 采用三种不同算子对图像进行锐
%化处理，在同一窗口输出图像；使用同态滤波器实现图像增强，输出图像。
% 读入 RGB 图像
rgbImage = imread('cat2.jpg'); % 替换为你的图像路径

% 显示原始 RGB 图像
figure(6);
subplot(2, 2, 1);
imshow(rgbImage);
title('原始图像');

% 1. 使用不同算子进行锐化处理
% Sobel 算子
sobelFilter = fspecial('sobel');
sharpenedSobel = imfilter(rgbImage, sobelFilter);
subplot(2, 2, 2);
imshow(sharpenedSobel);
title('Sobel 算子锐化');

% Prewitt 算子
prewittFilter = fspecial('prewitt');
sharpenedPrewitt = imfilter(rgbImage, prewittFilter);
subplot(2, 2, 3);
imshow(sharpenedPrewitt);
title('Prewitt 算子锐化');

% Laplacian 算子
laplacianFilter = fspecial('laplacian');
sharpenedLaplacian = imfilter(rgbImage, laplacianFilter);
subplot(2, 2, 4);
imshow(sharpenedLaplacian);
```

```
title('Laplacian 算子锐化');
sgtitle('锐化后图像');
%图像叠加
img1= imfuse(rgbImage, sharpenedSobel, 'blend');
img2= imfuse(rgbImage, sharpenedPrewitt, 'blend');
img3= imfuse(rgbImage, sharpenedLaplacian, 'blend');
figure(7);
subplot(2, 2, 1);
imshow(rgbImage);
title('原始图像');
subplot(2, 2, 2);
imshow(img1);
title('Sobel 算子锐化');
subplot(2, 2, 3);
imshow(img2);
title('Prewitt 算子锐化');
subplot(2, 2, 4);
imshow(img3);
title('Laplacian 算子锐化');
sgtitle('锐化图像与原始图像叠加');
%% 使用同态滤波器进行图像增强
%参数声明
rH = 1.5;
rL = 0.1;
c = 0.2;%介于 rH 和 rL 之间
D0 = 0.2;

image = imread('cat2.jpg');
[M, N] = size(image);
%取对数
img_log = log(double(image) + 1);

%平移到中心，判断语句代替指数计算
img_py = zeros(M, N);
for i = 1:M
```

```
    for j= 1:N
        if mod(i+j, 2) == 0
            img_py(i,j) = img_log(i, j);
        else
            img_py(i,j) = -1 * img_log(i, j);
        end
    end
end

% 对填充后的图像进行傅里叶变换
img_py_fft = fft2(img_py);

%同态滤波函数
img_tt = zeros(M, N);
deta_r = rH - rL;
D = D0^2;
m_mid=floor(M/2);%中心点坐标
n_mid=floor(N/2);

for i = 1:M
    for j =1:N
        dis = ((i-m_mid)^2+(j-n_mid)^2);
        img_tt(i, j) = deta_r * (1-exp((-c)*(dis/D))) + rL;
    end
end

%滤波
img_temp =   img_py_fft.*img_tt;

%反变换,取实部，绝对值
img_temp = abs(real(ifft2(img_temp)));

%指数化
img_temp = exp(img_temp) - 1;
```

```
%归一化处理
max_num = max(img_temp(:));
min_num = min(img_temp(:));
range = max_num - min_num;
img_after = zeros(M,N,'uint8');
for i = 1 : M
    for j = 1 : N
        img_after(i,j) = uint8(255 * (img_temp(i, j)-min_num) / range);
    end
end

img_after_RGB(:, :, 1) =img_after(:,1:276);
img_after_RGB(:, :, 2) =img_after(:,277:552);
img_after_RGB(:, :, 3) =img_after(:,553:828);

figure(9);
subplot(1,3,1), imshow(image), title('原图像');
subplot(1,3,2), imshow(img_after), title('同态滤波');
subplot(1,3,3), imshow(img_after_RGB), title('同态滤波叠加图像');
```