大连理工大学

本科实验报告

课程名称：数字图像处理技术实验

学院（系）：信息与通信工程学院

专　　业：通信工程

班　　级：电通 1301

学　　号：201383052

学生姓名：温思歆

2016 年 10 月 17 日

实验项目列表

序号	实验项目名称	学时	成　　绩			指导教师
			预习	操作	结果	
1	实验一 图像基本操作					
2	实验二 图像的边缘检测					
3	实验三 图像的 空域滤波和频域处理					
4	实验四 基于边缘 直方图的图像检索					
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						
15						
16						
17						
18						
总计	学分：					

大连理工大学实验预习报告

学院（系）：信通学院 专业：通信工程 班级：电通 1301

姓　　名：温思歆 学号：201383052 组：

实验时间：2016.10.10 实验室：创新园 C219 实验台：

指导教师签字：成绩：

实验一 图像基本操作

一、 实验目的和要求

1. 利用 matlab 获取图像进行读、写、显示等操作
2. 图像直方图均衡
3. 图像的点运算
4. 图像的几何变换

二、 实验用的 matlab 命令和例子

1.imread：读入各种图像文件，其语法格式[X ,MAP]=imread(filename,fmt)。

2. imwrite：输出图像，其语法格式为 Img=imwrite(X, MAP, filename, fmt)。

3.imshow：显示图像，其语法如下 imshow(I, n)或 imshow(I_BW)或 imshow(X, MAP)或 imshow(I_RGB)。

4. histeq：直方图均衡函数，基本调用格式为 J=histeq(I, n)，该函数返回原图像 I 经过直方图均衡化处理后的新图像 J，n 为指定的均衡化后的灰度级数，缺省值为 64。

5. 图像的点运算（图像相加，相减，线性变换，非线性变换（对数函数，幂次函数））

加法：对同一场景的多幅图像求平均值或者将一幅图像的内容叠加到另一幅图像上去，实现二次曝光。

减法：可以去除一幅图像中的所不需要的加性图像或者检测同一场景的两幅图像之间的变化。

6. imresize:图像缩放函数，其语法格式为：B = imresize(A, m, ‘method’)，返回原图像 A 的 m 倍放大的图像（m 小于 1 时效果是缩小）。这里参数’method’用于指定插值的方法，可选用的值为'nearest'（最邻近法），'bilinear'（双线性插值），'bicubic'（双三次插值），默认为'nearest'。

7. imrotate：图像旋转函数，其语法格式为：B = imrotate(A, angle, ‘method’)函数

imrotate 对图像进行旋转，参数’method’用于指定插值的方法，可选用的值为'nearest'（最邻近法），'bilinear'（双线性插值），'bicubic'（双三次插值），默认为'nearest'。一般说来旋转后的图像会比原图大，超出原图部分值为 0。

大连理工大学实验报告

学院（系）：信通学院 专业：通信工程 班级：电通 1301

姓　　名：温思歆 学号：201383052 组：

实验时间：2016.10.10 实验室：创新园 C219 实验台：

指导教师签字：成绩：

实验一 图像基本操作

一、 实验目的和要求

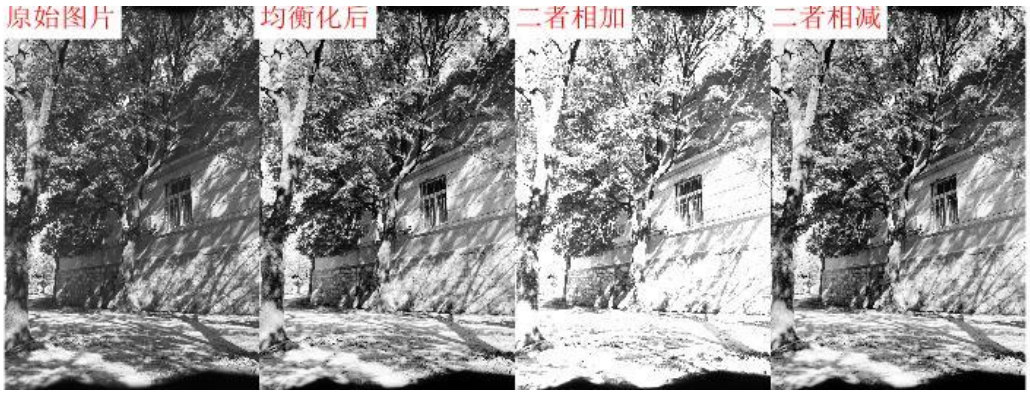
1. 利用 matlab 获取图像进行读、写、显示等操作
2. 图像直方图均衡
3. 图像的点运算
4. 图像的几何变换

二、实验题目、程序和结果

1. 图像与其直方图均衡后的图像相加
2. 图像与其直方图均衡后的图像相减

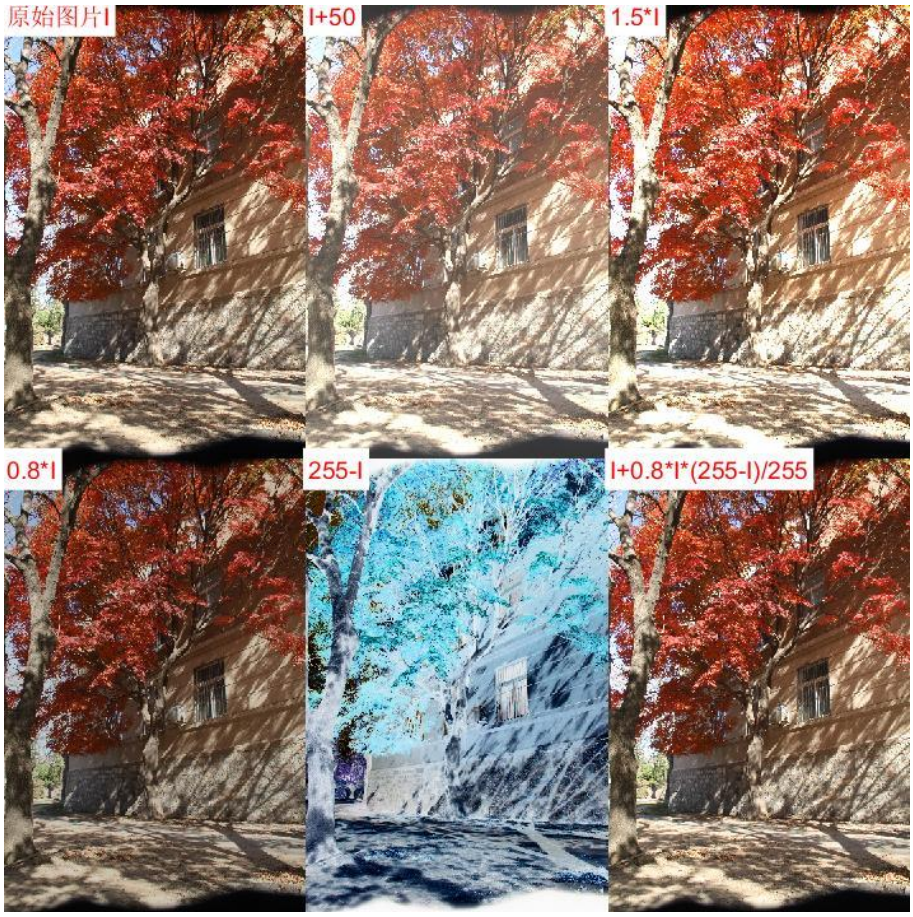
题目 1 和 2matlab 代码：

```
clc;clear
img=rgb2gray(imread('image/IMG_3696.jpg'));
histimg=histeq(img,256);
add_img=img+histimg;
sub_img=-img+histimg;

Img_show=[img histimg add_img sub_img];
figure(1);
imshow(img_show);
text(2+size(img,2)*0,100+size(img,1)*0,'原始图片','FontSize',14,'background','white');
text(2+size(img,2)*1,100+size(img,1)*0,'均衡化后','FontSize',14,'background','white');
text(2+size(img,2)*2,100+size(img,1)*0,'二者相加','FontSize',14,'background','white');
text(2+size(img,2)*3,100+size(img,1)*0,'二者相减' ,'FontSize',14,'background','white');
```



3. 图像线性变换和非线性变换

```
clc;clear
img=imread('image/IMG_3696.jpg');
D1=img+50;
D2=1.5*img;
D3=0.8*img;
D4=255-img;
D5=img+0.8*img.*(255-img)./255;

all=[img D1 D2; D3 D4 D5];
imshow(all);
text(20+size(img,2)*0,100+size(img,1)*0,'原始图片I' ,'FontSize',14,'background','white');
text(20+size(img,2)*1,100+size(img,1)*0,'I+50','FontSize',14,'background','white');
text(20+size(img,2)*2,100+size(img,1)*0,'1.5*I' ,'FontSize',14,'background','white');
text(20+size(img,2)*0,100+size(img,1)*1,'0.8*I', 'FontSize',14,'background','white');
text(20+size(img,2)*1,100+size(img,1)*1,'255-I','FontSize',14,'background','white');
text(20+size(img,2)*2,100+size(img,1)*1,'I+0.8*I*(255-I)/255','FontSize',14,'background','
white');
```



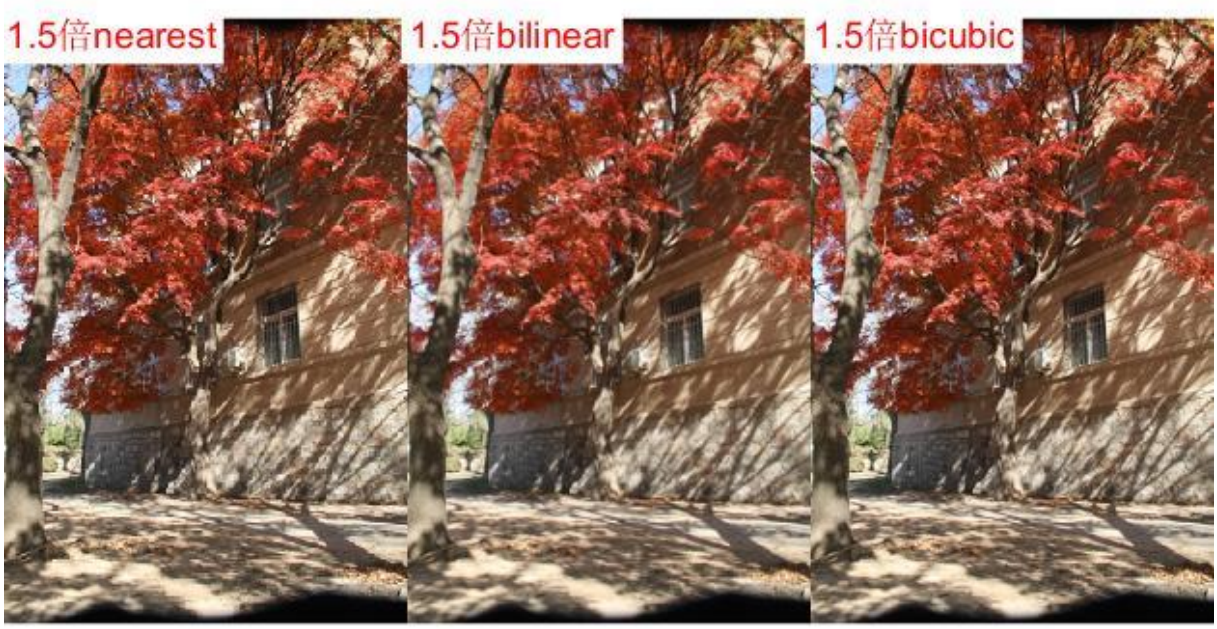
4. 将图像放大 1.5 倍，插值方法使用三种不同方法，显示放大后的图像，比较不同插值方法的结果有什么不同。

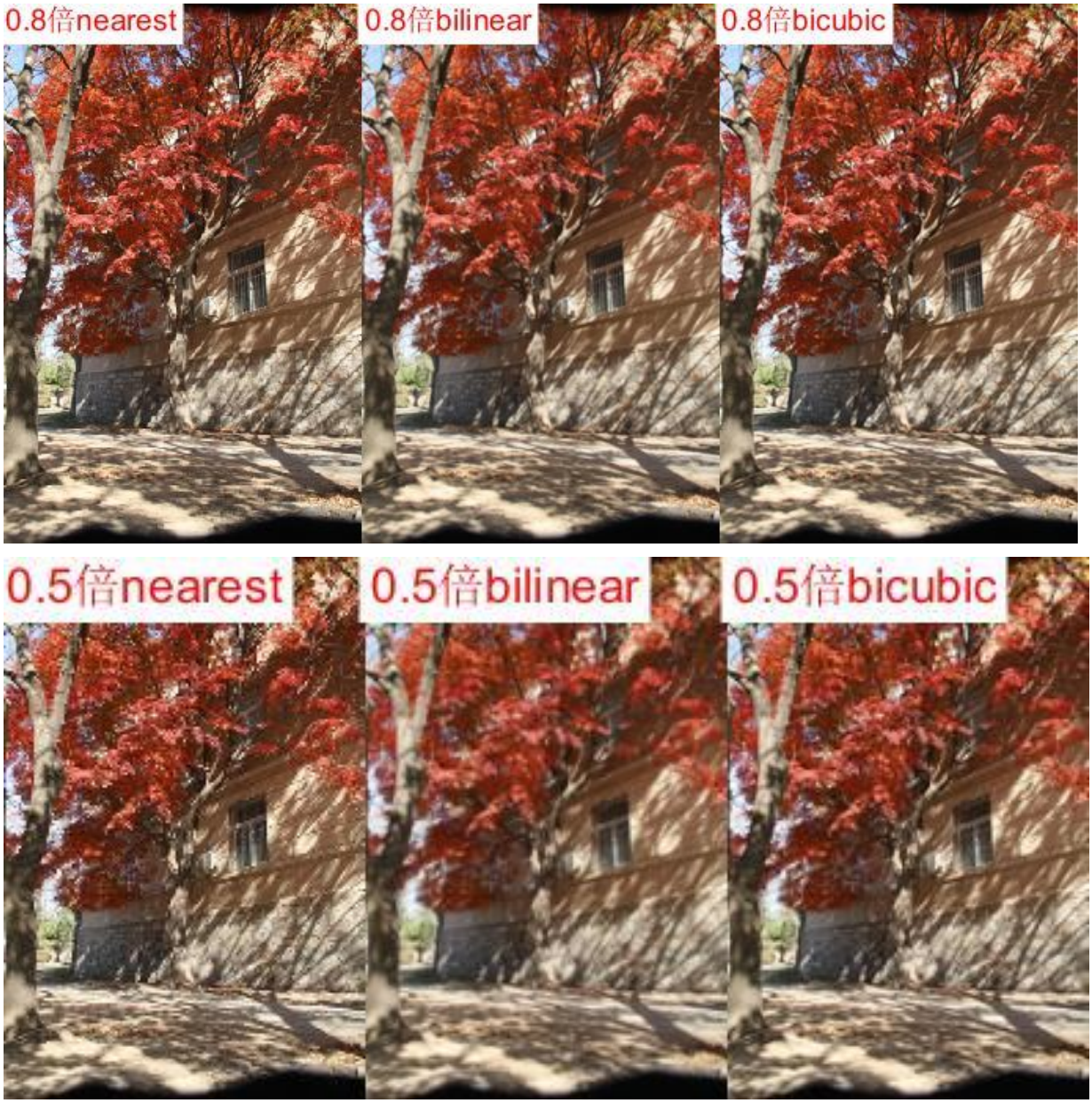
5. 图像缩小 0.8、0.5 倍，插值方法使用三种不同方法，显示并比较结果有什么差异。

题目 4 和 5matlab 代码：

```
clc;clear
img=imread('image/IMG_3696.jpg');
img=imresize(img,0.1);

img1=imresize(img,1.5,'nearest');
img2=imresize(img,1.5,'bilinear');
img3=imresize(img,1.5,'bicubic');
img_show=[img1 img2 img3];
figure(1);imshow(img_show);
text(2+size(img1,2)*0,15+size(img1,1)*0,'1.5倍nearest','FontSize',14,'background','white');
text(2+size(img1,2)*1,15+size(img1,1)*0,'1.5倍bilinear','FontSize',14,'background','white');
text(2+size(img1,2)*2,15+size(img1,1)*0,'1.5倍bicubic','FontSize',14,'background','white');
```

```
img1=imresize(img,0.8,'nearest');
img2=imresize(img,0.8,'bilinear');
img3=imresize(img,0.8,'bicubic');
img_show=[img1 img2 img3];
figure(2);imshow(img_show);
text(2+size(img1,2)*0,10+size(img1,1)*0,'0.8倍nearest','FontSize',14,'background','white');
text(2+size(img1,2)*1,10+size(img1,1)*0,'0.8倍bilinear','FontSize',14,'background','white');
text(2+size(img1,2)*2,10+size(img1,1)*0,'0.8倍bicubic','FontSize',14,'background','white');

img1=imresize(img,0.5,'nearest');
img2=imresize(img,0.5,'bilinear');
img3=imresize(img,0.5,'bicubic');
img_show=[img1 img2 img3];
figure(3);imshow(img_show);
text(2+size(img1,2)*0,10+size(img1,1)*0,'0.5倍nearest','FontSize',14,'background','white');
text(2+size(img1,2)*1,10+size(img1,1)*0,'0.5倍bilinear','FontSize',14,'background','white');
text(2+size(img1,2)*1,10+size(img1,1)*0,'0.5倍bicubic','FontSize',14,'background','white');
```



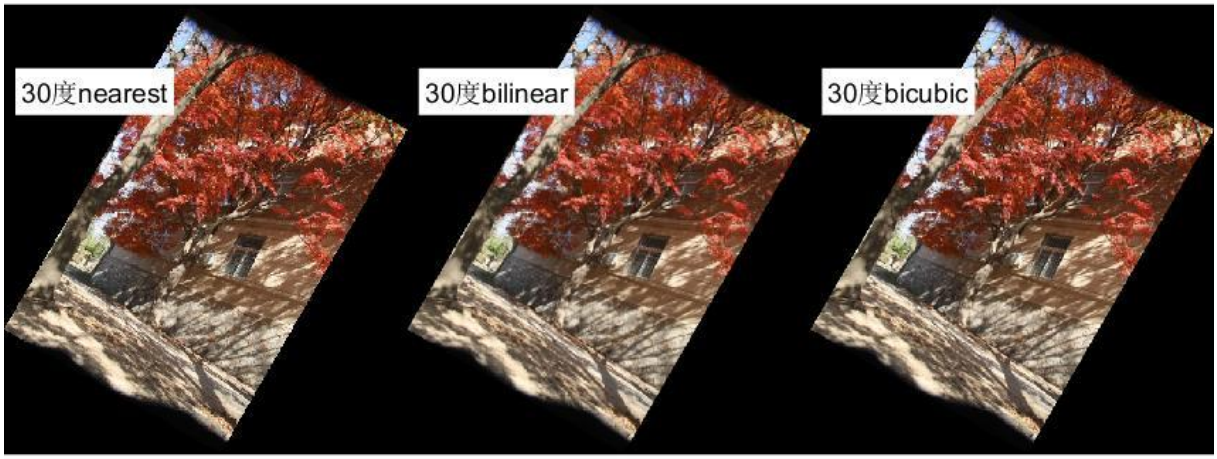


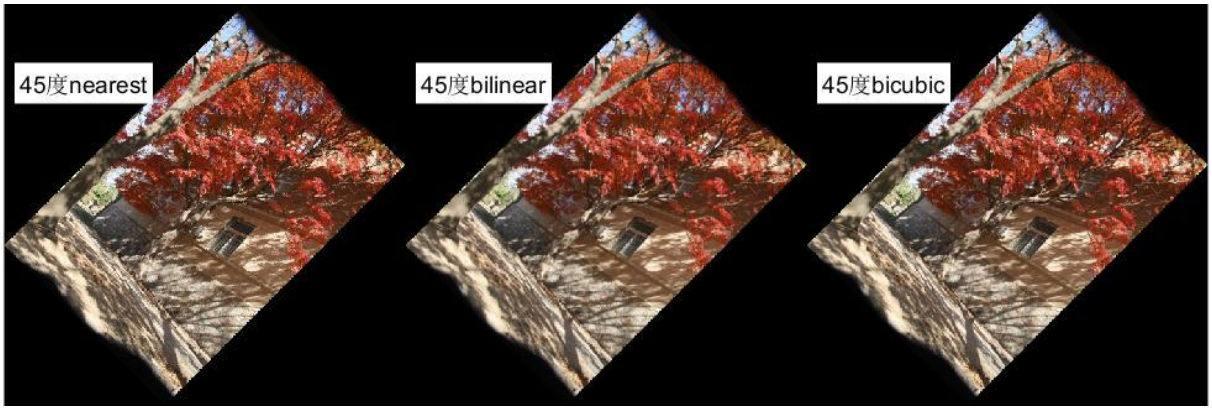
结果分析：nearest 方法最粗糙，可以看到一个个像素块；bilinear 方法效果有一些模糊；bicubic 效果最好。

6. 图像分别顺时针旋转 30 度、45 度，插值方法使用三种不同方法，显示旋转后的图像并比较结果有什么不同。

```
clc;clear
img=imread('image/IMG_3696.jpg');
img=imresize(img,0.1);

img1=imrotate(img,360-30,'nearest');
img2=imrotate(img,360-30,'bilinear');
```

```
    img3=imrotate(img,360-30,'bicubic');
    img_show=[img1 img2 img3];
    figure(1);imshow(img_show);
text(20+size(img1,2)*0,100+size(img1,1)*0,'30度nearest','FontSize',14,'background','white');
text(20+size(img1,2)*1,100+size(img1,1)*0,'30度bilinear','FontSize',14,'background','white');
text(20+size(img1,2)*2,100+size(img1,1)*0,'30度bicubic','FontSize',14,'background','white');

    img1=imrotate(img,360-45,'nearest');
    img2=imrotate(img,360-45,'bilinear');
    img3=imrotate(img,360-45,'bicubic');
    img_show=[img1 img2 img3];
figure(2);imshow(img_show);
text(20+size(img1,2)*0,100+size(img1,1)*0,'45度nearest','FontSize',14,'background','white');
text(20+size(img1,2)*1,100+size(img1,1)*0,'45度bilinear','FontSize',14,'background','white');
text(20+size(img1,2)*2,100+size(img1,1)*0,'45度bicubic','FontSize',14,'background','white');
```





分析：bilinear 效果最差。

三、实验分析和总结

1. 图像均衡函数 histeq 是对灰度图做处理，不能对彩色图片处理，所以

需要用 rgb2gray 函数进行转换。

2. 两图像直接相加可能会导致过曝光，即像素值超过了显示范围，故理论上应该相加后取平均。
3. 对图像进行加减过程要注意数据形式是 double 还是 uint8_t，否则很容易就出现显示错误。
4. 将图像用[img1 img2]合在一起显示出来较方便，也减少了一些不必要的空白区，但添加标注时要每次注意找出图像的具体位置。
5. 对图像进行插值时应该用分辨率较低的图像，否则三种插值效果的区别不明显，故本次实验采用老师给的图片后先统一用 imresize(img,0.1) 函数来降低分辨率。
6. 用 imrotate 函数时默认角度是逆时针旋转，但题目要求是顺时针旋转，故需要用 360-angle 才可以。

大连理工大学实验预习报告

学院（系）：信通学院 专业：通信工程 班级：电通 1301

姓　　名：温思歆 学号：201383052 组：

实验时间：2016.10.10 实验室：创新园 C219 实验台：

指导教师签字：成绩：

实验二 图像的边缘检测

一、 实验目的和要求

1. 理解图像边缘提取的基本概念；
2. 熟悉进行边缘提取的基本方法；
3. 掌握用 MATLAB 语言进行图像边缘提取的方法。

二、 实验用的 matlab 命令和例子

边缘检测技术对于处理数字图像非常重要，因为边缘是所要提取目标和背景的分界线，提取出边缘才能将目标和背景区分开来。在图像中，边界表明一个特征区域的终结和另一个特征区域的开始，边界所分开区域的内部特征或属性是一致的，而不同的区域内部的特征或属性是不同的，边缘检测正是利用物体和背景在某种图像特性上的差异来实现的，这些差异包括灰度，颜色或者纹理特征。边缘检测实际上就是检测图像特征发生变化的位置。由于噪声和模糊的存在，检测到的边界可能会变宽或在某些点处发生间断，因此，边界检测包括两个基本内容：首先抽取出反映灰度变化的边缘点，然后剔除某些边界点或填补边界间断点，并将这些边缘连接成完整的线。边缘检测的方法大多数是基于方向导数掩模求卷积的方法。

对于数字图像，应该采用差分运算代替求导，相对应的一阶差分为：

$$\Delta_x f(i,j) = f(i,j) - f(i-1,j)$$
$$\Delta_y f(i,j) = f(i,j) - f(i,j-1)$$

方向差分为：

$$\Delta_\alpha f(i,j) = \Delta_x f(i,j)\cos\alpha + \Delta_y f(i,j)\sin\alpha$$

函数 f 在某点的方向导数取得最大值的方向是 方向导数的最大值称为梯度模。利用梯度模算子来检测边缘是一种很好的方法，它不仅具有位移不变性，还具有各向同性。

为了运算简便，实际中采用梯度模的近似形式，另外，还有一些常用的算子，如 Roberts 算子和 Sobel 算子。其中，由于 Sobel 算子是滤波算子的形式，用于提取边缘。我们可以利用快速卷积函数，简单有效，因此应用很广泛。

MATLAB 的图像处理工具箱中提供的 edge 函数可以实现检测边缘的功能，其语法格式如下：

BW = edge(I, ‘sobel’)

BW = edge(I, ‘sobel’, direction)

BW = edge(I, ‘roberts’)

BW = edge(I, ‘log’)

这里 BW = edge(I, ‘sobel’)采用 Sobel 算子进行边缘检测。BW = edge(I, ‘sobel’, direction)可以指定算子方向，即：

direction = ‘horizontal’，为水平方向；

direction = ‘vertical’，为垂直方向；

direction = ‘both’，为水平和垂直两个方向。

BW = edge(I, ‘roberts’)和 BW = edge(I, ‘log’)分别为用 Roberts 算子和拉普拉斯高斯算子进行边缘检测。

图像无噪声时，可用 Roberts 算子；Prewitt 和 Sobel 算子同时具有平均，即抑制噪声作用；对阶跃状边缘，Roberts 得到的边缘宽度≥1 个像素，Prewitt 和 Sobel 算子得到的边缘宽度≥2 个像素。

在利用 edge 函数进行相应的算子边缘检测的时候，各算子的差别非常微小，不过由相应的参数，三个算子分别为 0.08、0.05、0.04 可以知道，Sobel 算子在边缘检测中最为敏感，及在同一条件下它的处理效果应该最好。

大连理工大学实验报告

学院（系）：信通学院 专业：通信工程 班级：电通 1301

姓　　名：温思歆 学号：201383052 组：

实验时间：2016.10.10 实验室：创新园 C219 实验台：

指导教师签字： 成绩：

实验二 图像的边缘检测

一、 实验目的和要求

1. 理解图像边缘提取的基本概念；
2. 熟悉进行边缘提取的基本方法；
3. 掌握用 MATLAB 语言进行图像边缘提取的方法。

二、实验题目、程序和结果

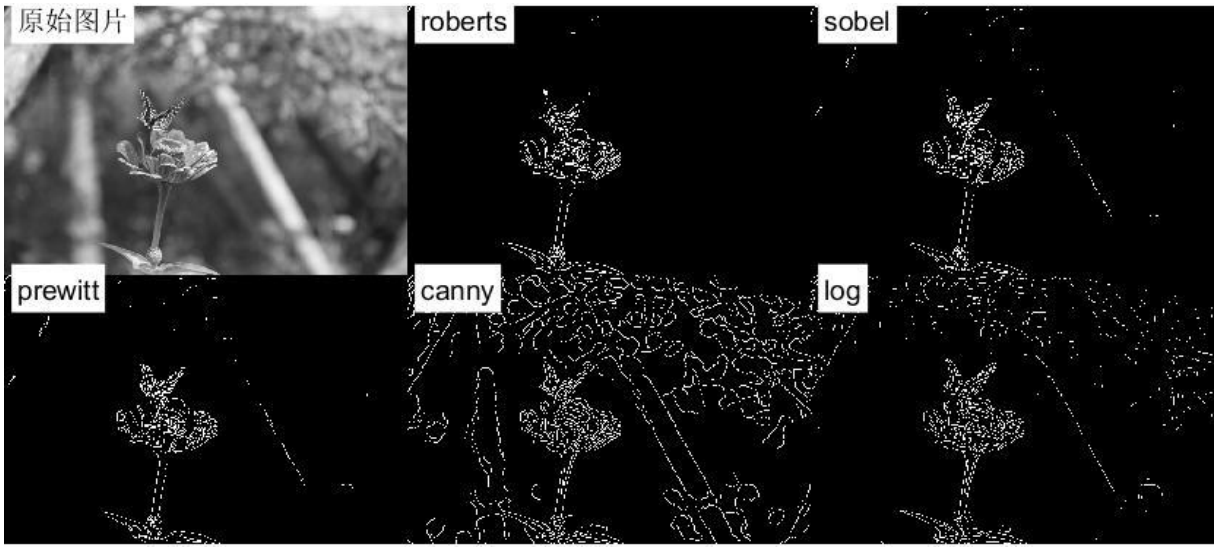
1.分别用 Roberts、Sobel 和 Prewitt 算子对图像进行边缘检测。比较三种算子处理的结果。

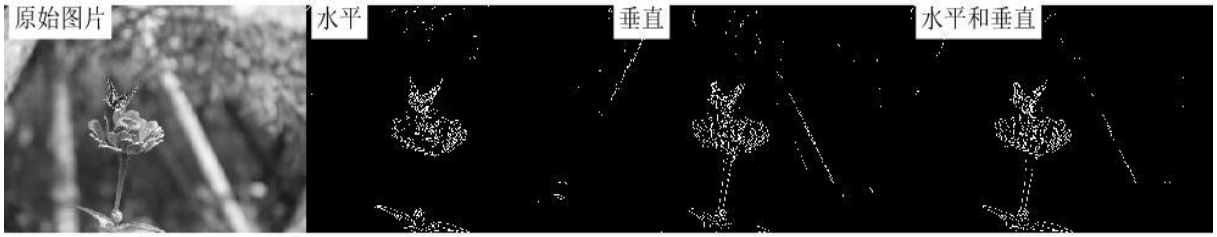
2. 用不同方向（‘水平’、‘垂直’、‘水平和垂直’）的 Sobel 算子对图像进行边缘检测。比较三种情况的结果。

题目 1 和题目 2 代码：

```
clc;clear
img=rgb2gray(imresize(imread('image\实验一二/IMG_4441.jpg'),0.25));
img=im2double(img);
img1=edge(img,'roberts');
img2=edge(img,'sobel');
img3=edge(img,'prewitt');
img4=edge(img,'canny');
img5=edge(img,'log');

img_show=[img img1 img2;img3 img4 img5];
imshow(img_show);
text(20+size(img,2)*0,20+size(img,1)*0,'原始图片','FontSize',14,'background','white');
text(20+size(img,2)*1,20+size(img,1)*0,'roberts','FontSize',14,'background','white');
text(20+size(img,2)*2,20+size(img,1)*0,'sobel','FontSize',14,'background','white');
text(20+size(img,2)*0,20+size(img,1)*1,'prewitt','FontSize',14,'background','white');
```

```
text(20+size(img,2)*1,20+size(img,1)*1,'canny','FontSize',14,'background','white');
text(20+size(img,2)*2,20+size(img,1)*1,'log','FontSize',14,'background','white');

img6=edge(img,'sobel','horizontal');
img7=edge(img,'sobel','vertical');
img8=edge(img,'sobel','both');
img_dire=[img img6 img7 img8];
figure;imshow(img_dire);
text(20+size(img,2)*0,20+size(img,1)*0,'原始图片','FontSize',14,'background','white');
text(20+size(img,2)*1,20+size(img,1)*0,'水平','FontSize',14,'background','white');
text(20+size(img,2)*2,20+size(img,1)*0,'垂直','FontSize',14,'background','white');
text(20+size(img,2)*3,20+size(img,1)*0,'水平和垂直','FontSize',14,'background','white');
```





结果分析：由运行结果，sobel 算子和 prewitt 算子的边缘检测效果最好。roberts 丢失了一些信息，这三种对前景的提取都很好，而 canny 和 log 对前景的提取就不理想。

水平边缘检测主要检测出水平边缘；垂直边缘检测主要检测出垂直边缘，而水平和垂直方向的边缘检测对信息的保持最好。

3、 自编程序，实现一种利用模版进行边缘检测处理，也就是调用 filter2

函数利用模版对图像进行滤波即可。

```
clc;clear
img=rgb2gray(imresize(imread('image\实验一二/IMG_4441.jpg'),0.25));
img=im2double(img);

cof1=[1 2 1;2 -12 2;1 2 1];
img1=filter2(cof1,img);
cof2=[1 1 1;1 -8 1;1 1 1];
img2=filter2(cof2,img);

img_show=[img img1 img2];
imshow(img_show);
text(20+size(img,2)*0,20+size(img,1)*0,'原始图片','FontSize',14,'background','white');
text(20+size(img,2)*1,20+size(img,1)*0,'模板[1 2 1;2 -12 2;1 2
1]','FontSize',14,'background','white');
text(20+size(img,2)*2,20+size(img,1)*0,'模板[1 1 1;1 -8 1;1 1
1]','FontSize',14,'background','white');
```

对边缘的检测效果较好，把前景细节信息都保持下来了。

三、实验分析和总结

1. 通过 doc+函数名可以打开该函数的官方文档，故我用 doc edge 命令后发现 edge 函数的参数还有 canny、log、prewitt、roberts、sobel 等，故一并处理后显示出来，比较其不同的结果。
2. 自主设计边缘检测处理的模板时，依照理论上学到矩阵和为 0 时是高通滤波可用检测边缘，设计了两组模板[1 2 1;2 -12 2;1 2 1]、[1 1 1;1 -8 1;1 1 1]，效果出奇的非常好，验证了理论与实践相结合的作用。
3. 实际上 filter2 和 conv2 函数是一样的，通过 edit filter2 即可看到 filter2 是调用了 conv2 函数。

大连理工大学实验预习报告

学院（系）：信通学院 专业：通信工程 班级：电通 1301

姓　　名：温思歆 学号：201383052 组：

实验时间：2016.10.10 实验室：创新园 C219 实验台：

指导教师签字：成绩：

实验三 图像的空域滤波和频域处理

一、 实验目的和要求

1、理解图像空域滤波的基本定义及目的；

2、掌握图像空域滤波的基本原理及方法；

3、掌握用 MATLAB 语言实现图像的空域滤波的方法。

4、理解离散傅立叶变换的基本原理；

5、掌握应用 MATLAB 语言进行 FFT 及逆变换的方法。

二、 实验用的 matlab 命令和例子

1、均值滤波：均值滤波是在空间域对图像进行平滑处理的一种方法，易于实现，效果也挺好。设噪声 η (m,n)是加性噪声，其均值为 0，方差（噪声功率）为 σ 2，而且噪声与图像 f(m,n)不相关。其有噪声的图像 f’(m, n)为：

$$f'(m,n)=f(m,n)+\eta(m,n) \tag{3.1}$$

经均值滤波处理后的图像 g(m, n)为：

$$g(m,n)=\frac{1}{N}\sum_{(i,j)\in s}\sum f'(i,j)=\frac{1}{N}\sum_{(i,j)\in s}\sum[f(i,j)+\eta(i,j)]=\frac{1}{N}\sum_{(i,j)\in s}\sum f(i,j)+\frac{1}{N}\sum_{(i,j)\in s}\sum\eta(i,j) \tag{3.2}$$

其中 s 是（m, n）点的领域内的点集。

除了对噪声有上述假定之外，该算法还基于这样一种假设：图像是由许多灰度值相近的小块组成。这个假设大体上反映了许多图像的结构特征。（3.2）式表达的算法是由某像素邻域内各点灰度值的平均值来代替该像素原来的灰度值。

可用模块反映邻域平均算法的特征。对于四点邻域和八点邻域，可分别由下述摸板表征：

$$M_1=\begin{pmatrix}0 & \frac{1}{5} & 0\\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5}\\ 0 & \frac{1}{5} & 0\end{pmatrix}=\frac{1}{5}\begin{pmatrix}0 & 1 & 0\\ 1 & 1 & 1\\ 0 & 1 & 0\end{pmatrix} \tag{4.3}$$

$$M_2=\begin{pmatrix}\frac{1}{9} & \frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9}\end{pmatrix}=\frac{1}{9}\begin{pmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{pmatrix} \tag{4.4}$$

模版沿水平和垂直两个方向逐点移动，相当于用这样一个模块与图像进行卷积运算，从而平滑了整幅图像。模版内各系数和为 1，用这样的模版处理常数图像时，图像没有变化；对一般图像处理后，整幅图像灰度的平均值可不变。

2、中值滤波：中值滤波是一种非线性处理技术，能抑制图像中的噪声。它是基于图像的这样一种特性：噪声往往以孤立的点的形式出现，这些点对应的象素很少，而图像则是由像素数较多、面积较大的小块构成。

在一维的情况下，中值滤波器是一个含有奇数个像素的窗口。在处理之后，位于窗口正中的像素的灰度值，用窗口内各像素灰度值的中值代替。例如若窗口长度为 5，窗口中像素的灰度值为 80、90、200、110、120，则中值为 110，因为按小到大（或大到小）排序后，第三位的值是 110。于是原理的窗口正中的灰度值 200 就由 110 取代。如果 200 是一个噪声的尖峰，则将被滤除。然而，如果它是一个信号，则滤波后就被消除，降低了分辨率。因此中值滤波在某些情况下抑制噪声，而在另一些情况下却会抑制信号。中值滤波很容易推广到二维的情况。二维窗口的形式可以是正方形、近似圆形的或十字形的。在图像增强的具体应用中，中值滤波只能是一种抑制噪声的特殊工具，在处理中应监视其效果，以决定最终是否采用这种方案。实施过程中的关键问题是探讨一些快速算法。

MATLAB 图像处理工具箱提供了基于卷积的图像滤波函数 filter2。filter2 的语法格式为：

Y = filter2(h,X)

其中 Y = filter2(h,X)返回图像 X 经算子 h 滤波后的结果，默认返回图像 Y 与输入图像 X 大小相同。

fspecial 函数用于创建预定义的滤波算子，其语法格式为：

h = fspecial(type)

h = fspecial(type,parameters)

参数 type 指定算子类型，parameters 指定相应的参数，具体格式为：

type='average'，为均值滤波，参数 parameters 为 n，代表模版尺寸，用向量表示，默认值为[3,3]。

type='gaussian'，为高斯低通滤波器，参数 parameters 有两个，n 表示模版尺寸，默认值为[3,3]，sigma 表示滤波器的标准差，单位为像素，默认值为 0.5。

type= 'laplacian'，为拉普拉斯算子，参数 parameters 为 alpha，用于控制拉普拉斯算子的形状，取值范围为[0,1]，默认值为 0.2。

type= 'log'，为拉普拉斯高斯算子，参数 parameters 有两个，n 表示模版尺寸，默认值为[3,3]，sigma 为滤波器的标准差，单位为像素，默认值为 0.5

type= 'prewitt'，为 prewitt 算子，用于边缘增强，无参数。

type= 'sobel'，为著名的 sobel 算子，用于边缘提取，无参数。

type= 'unsharp'，为对比度增强滤波器，参数 alpha 用于控制滤波器的形状，范围为[0,1]，默认值为 0.2。

3、傅立叶变换的基本知识

在图像处理的广泛应用领域中，傅立叶变换起着非常重要的作用，具体表现在包括图像分析、图像增强及图像压缩等方面。

假设 f（x, y）是一个离散空间中的二维函数，则该函数的二维傅立叶变换的定义如下：F（p, q）称为 f（m, n）的离散傅立叶变换系数。这个式子表明，函数 f（m, n）可以用无数个不同频率的复指数信号和表示，而在频率（w1，w2）处的复指数信号的幅度和相位是 F（w1，w2）。

将傅立叶变换的结果进行可视化的另一种方法是用图像的方式显示变换结果的对数幅值，如图所示。

4、MATLAB 提供的快速傅立叶变换函数

（1） fft2

fft2 函数用于计算二维快速傅立叶变换，其语法格式为：

B= fft2(I)

B = fft2(I)返回图像 I 的二维 fft 变换矩阵，输入图像 I 和输出图像 B 大小相同。

例如，计算图像的二维傅立叶变换，并显示其幅值的结果，如图所示，其命令格式如下

load imdemos saturn2

imshow(saturn2)

B = fftshift(fft2(saturn2));

imshow(log(abs(B)),[],‘notruesize’)

（2） fftshift

MATLAB 提供的 fftshift 函数用于将变换后的图像频谱中心从矩阵的原点移到矩阵的中心，其语法格式为：

B = fftshift(I)

对于矩阵 I，B = fftshift(I)将 I 的一、三象限和二、四象限进行互换。

（3） ifft2

ifft2 函数用于计算图像的二维傅立叶反变换，其语法格式为：

B = ifft2(I)

B = ifft2(I)返回图像 I 的二维傅立叶反变换矩阵，输入图像 I 和输出图像 B 大小相同。其语法格式含义与 fft2 函数的语法格式相同，可以参考 fft2 函数的说明。

大连理工大学实验报告

学院（系）：信通学院 专业：通信工程 班级：电通 1301

姓　　名：温思歆 学号：201383052 组：

实验时间：2016.10.10 实验室：创新园 C219 实验台：

指导教师签字： 成绩：

实验三 图像的空域滤波和频域处理

一、 实验目的和要求

1、理解图像空域滤波的基本定义及目的；

2、掌握图像空域滤波的基本原理及方法；

3、掌握用 MATLAB 语言实现图像的空域滤波的方法。

4、理解离散傅立叶变换的基本原理；

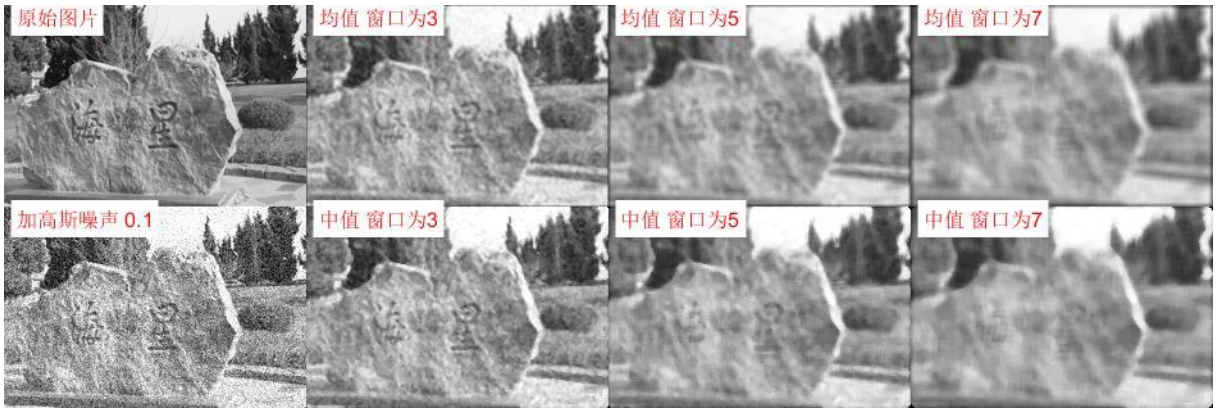
5、掌握应用 MATLAB 语言进行 FFT 及逆变换的方法。

二、实验题目、程序和结果

1、分别采用不同大小的模板对加有高斯噪声的图像进行均值滤波和中值滤波，在一个图形显示窗口中显示原图像、加有噪声的图像，均值滤波的图像以及中值滤波的图像。比较结果。

```
clc;clear
img=imresize(imread('image\实验一二/IMG_3573.jpg'),0.1);
img=im2double(rgb2gray(img));
noise_img=imnoise(img,'gaussian',0.1);

% 均值滤波
cof1=fspecial('average',3);
img1=filter2(cof1,noise_img);
cof2=fspecial('average',5);
img2=filter2(cof2,noise_img);
cof3=fspecial('average',7);
img3=filter2(cof3,noise_img);

%中值滤波
img4=medfilt2(noise_img,[3 3]);
img5=medfilt2(noise_img,[5 5]);
img6=medfilt2(noise_img,[7 7]);
```

```
img_show=[img img1 img2 img3;noise_img img4 img5 img6];
figure(1);
imshow(img_show);
text(10+size(img,2)*0,10+size(img,1)*0,'原始图片','FontSize',12,'background','white');
text(10+size(img,2)*1,10+size(img,1)*0,'均值  窗口为3','FontSize',12,'background','white');
text(10+size(img,2)*2,10+size(img,1)*0,'均值  窗口为5','FontSize',12,'background','white');
text(10+size(img,2)*3,10+size(img,1)*0,'均值  窗口为7','FontSize',12,'background','white');

text(10+size(img,2)*0,10+size(img,1)*1,'加高斯噪声  0.1','FontSize',12,'background','white');
text(10+size(img,2)*1,10+size(img,1)*1,'中值  窗口为3','FontSize',12,'background','white');
text(10+size(img,2)*2,10+size(img,1)*1,'中值  窗口为5','FontSize',12,'background','white');
text(10+size(img,2)*3,10+size(img,1)*1,'中值  窗口为7','FontSize',12,'background','white');
```



2、（选做内容）不调用工具箱的函数，自编程序，实现均值滤波和中值滤波。

主函数：

```
clc;clear;dbstop if error
img=imresize(imread('image\实验一二/IMG_3573.jpg'),0.1);
img=im2double(rgb2gray(img));
noise_img=imnoise(img,'gaussian',0.1);
% img=[8 6 13 9;1 13 1 15;5 4 7 7;5 10 3 7];
%  均值滤波
cof1=fspecial('average',3);
img1=maverfilt(cof1,noise_img);
cof2=fspecial('average',5);
img2=maverfilt(cof2,noise_img);
cof3=fspecial('average',7);
img3=maverfilt(cof3,noise_img);
%中值滤波
img4=mmediafilt(noise_img,[3 3]);
img5=mmediafilt(noise_img,[5 5]);
img6=mmediafilt(noise_img,[7 7]);
img_show=[img img1 img2 img3;noise_img img4 img5 img6];
figure(1);
imshow(img_show);
```

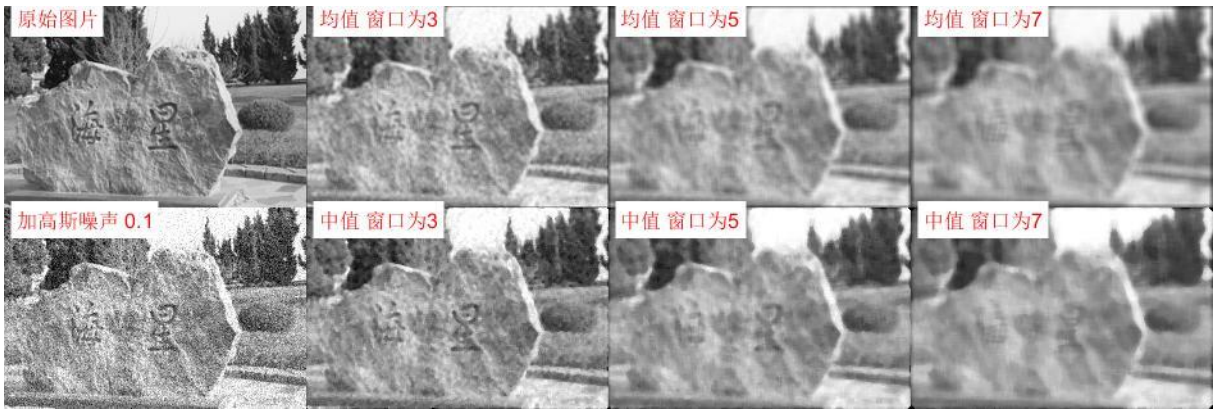
```
text(10+size(img,2)*0,10+size(img,1)*0,'原始图片','FontSize',12,'background','white');
text(10+size(img,2)*1,10+size(img,1)*0,'均值  窗口为3','FontSize',12,'background','white');
text(10+size(img,2)*2,10+size(img,1)*0,'均值  窗口为5','FontSize',12,'background','white');
text(10+size(img,2)*3,10+size(img,1)*0,'均值  窗口为7','FontSize',12,'background','white');

text(10+size(img,2)*0,10+size(img,1)*1,'加高斯噪声  0.1','FontSize',12,'background','white');
text(10+size(img,2)*1,10+size(img,1)*1,'中值  窗口为3','FontSize',12,'background','white');
text(10+size(img,2)*2,10+size(img,1)*1,'中值  窗口为5','FontSize',12,'background','white');
text(10+size(img,2)*3,10+size(img,1)*1,'中值  窗口为7','FontSize',12,'background','white');
```

中值滤波函数：

```
function new_img=mmediafilt(varargin)
a =varargin{1};mn=varargin{2};
x=mn(1);y=mn(2);
img=zeros(size(a,1)+floor(x/2)*2,size(a,2)+floor(y/2)*2);
img(round(x/2):size(a,1)+round(x/2)-1,round(y/2):size(a,2)+round(y/2)-1)=a;
for i=1:size(a,1)
    for j=1:size(a,2)
        tmp=img(i:i+x-1,j:j+y-1);
        tmp2=reshape(tmp,size(tmp,1)*size(tmp,2),1);
        s=sort(tmp2);
        new_img(i,j)=s(floor(size(tmp2,1)/2));
    end
end
```

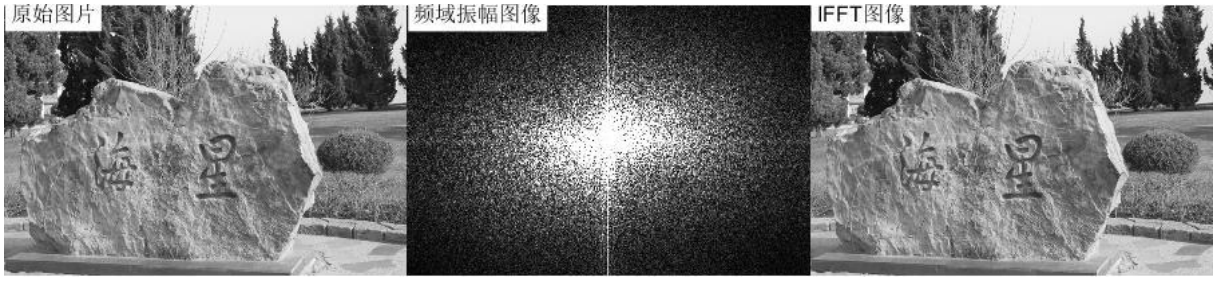
均值滤波函数：

```
function new_img=maverfilt(varargin)
a =varargin{2};mn=varargin{1};
x=size(mn,1);y=size(mn,2);
img=zeros(size(a,1)+floor(x/2)*2,size(a,2)+floor(y/2)*2);
img(round(x/2):size(a,1)+round(x/2)-1,round(y/2):size(a,2)+round(y/2)-1)=a;

for i=round(x/2):size(a,1)+round(x/2)-1
    for j=round(y/2):size(a,2)+round(y/2)-1
        tmp(i,j)=0;
        for k=1:x
            for n=1:y
                tmp(i,j)=tmp(i,j)+ img(i-round(x/2)+k,j-round(y/2)+n);
            end
        end
    end
end
tmp=tmp/(x*y);
new_img=tmp(round(x/2):size(a,1)+round(x/2)-1,round(y/2):size(a,2)+round(y/2)-1);
```



3、读取一幅灰度图像，显示这幅图像，对图像作傅立叶变换，显示频域振幅图像。作傅立叶逆变换，显示图像，看是否与原图像相同。

```
clc;clear
img=imresize(imread('image\实验一二/IMG_3573.jpg'),0.25);
img=im2double(rgb2gray(img));

fft_img=fftshift(fft2(img));
amp_img=abs(fft_img)/100;
ifft_img=ifft2(ifftshift(fft_img));

img_show=[img amp_img ifft_img];
imshow(img_show);
text(10+size(img,2)*0,10+size(img,1)*0,'原始图片','FontSize',12,'background','white');
text(10+size(img,2)*1,10+size(img,1)*0,'频域振幅图像','FontSize',12,'background','white');
text(10+size(img,2)*2,10+size(img,1)*0,'IFFT图像','FontSize',12,'background','white');
```

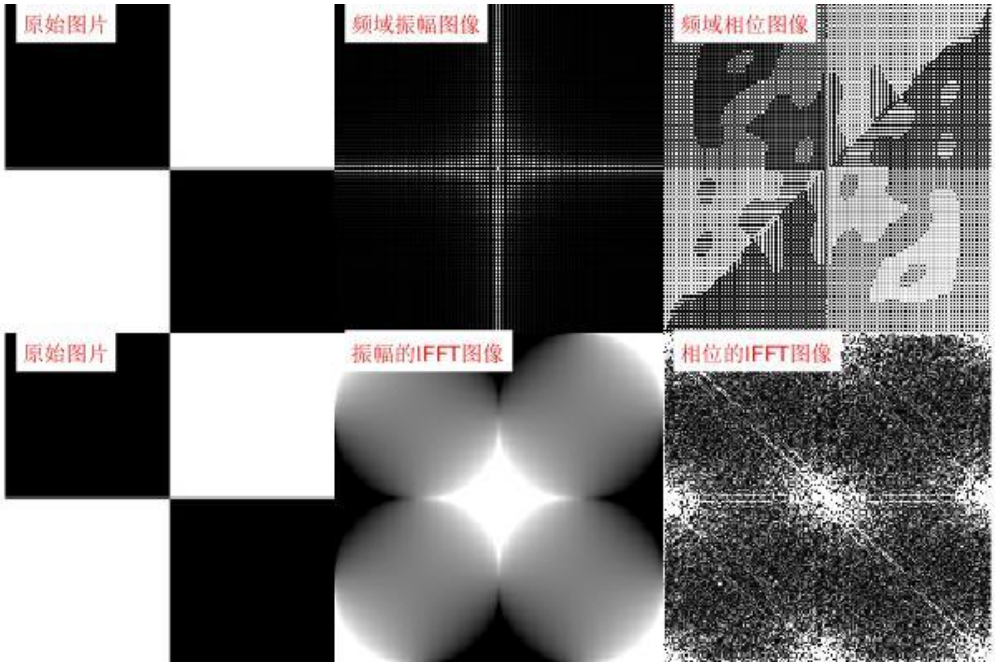


做逆变换后可以恢复原来图像。

4、（选做内容）：对一幅图像作傅立叶变换，显示一幅频域图像的振幅分布图和相位分布图，分别对振幅分布和相位分布作傅立叶逆变换，观察两幅逆变换后的图像，体会频域图像中振幅与位相的作用。

```
clc;clear;
img=imresize(imread('image\icon.jpg'),1);
img=im2double((img));
```

```
fft_img=fftshift(fft2(img));
angle_img=360*angle(fft_img)/pi;
map_img=abs(fft_img);

ifft_angle=ifft2(ifftshift(angle_img));
ifft_map=ifft2(ifftshift(map_img));

img_show=[img (map_img)/100 mat2gray(angle_img);img ifft_map abs(ifft_angle)];
imshow(img_show);
text(10+size(img,2)*0,10+size(img,1)*0,'原始图片','FontSize',10,'background','white');
text(10+size(img,2)*1,10+size(img,1)*0,'频域振幅图像','FontSize',10,'background','white');
text(10+size(img,2)*2,10+size(img,1)*0,'频域相位图像','FontSize',10,'background','white');
text(10+size(img,2)*0,10+size(img,1)*1,'原始图片','FontSize',10,'background','white');
text(10+size(img,2)*1,10+size(img,1)*1,'振幅的IFFT图像','FontSize',10,'background','white');
text(10+size(img,2)*2,10+size(img,1)*1,'相位的IFFT图像','FontSize',10,'background','white');
```



分析：振幅描述图像灰度的亮度，而相位则表示方向，只有两者结合起来才能恢复出原始图像。

三、实验分析和总结

1. 因为噪声一般是高频信号，故可用低通滤波来得到标准图像，而均值滤波和中值滤波正好就是低通滤波，故可以用来除噪。但是滤波参数要选好，避免图像原来的高频细节丢失，如选用[7 7]窗口时得到图像已非常模糊，丢失了大量信息。
2. 自主设计中值和均值滤波函数时最应该注意的就是图像边界的处理问题，题目没有具体要求，故我采用了最简单的补零操作。另外需要注意不要数组越界，再加上简单的处理就可设计出这两个函数。

3. 对图像做 ifft2 时需要注意原来的 fft2 有没有加上 fftshift 操作，若有则需要加上 ifftshift 将频域移动至标准模式，这样才能获得正确结果。
4. 对图像做 fft2 处理后要显示时需要注意此时振幅和相位都是 double 类型，且数值较大，相位值还有可能是负数，故需用 mat2gray 将数据变换到[0 1]区间，但是振幅显示不直观，故采用总体除以 100 的操作显示出来。
5. 仅仅利用相位或幅值信息来做 ifft2 处理时不能得到原图，但结合 ifft2 后的振幅图和相位图可知，幅值反应的是灰度大小而相位则保存了图像大量的信息。参阅网上博客 http://blog.csdn.net/on2way/article/details/46981825 可知，实际上相位更重要。这个实验让我深刻理解了图像的幅值和相位的作用，非常有意义，受益匪浅。

大连理工大学实验预习报告

学院（系）：信通学院 专业：通信工程 班级：电通 1301

姓　　名：温思歆 学号：201383052 组：

实验时间：2016.10.10 实验室：创新园 C219 实验台：

指导教师签字：成绩：

实验四 基于边缘直方图的图像检索

一、 实验目的和要求

1. 掌握用 matlab 进行图像检索的方法。

二、 实验用的 matlab 命令和例子

通常情况下，人类可以在只用图像的轮廓而不借助图像的颜色以及纹理信息的情况下就可以对一幅图像中的物体进行分类和识别。因而，研究图像轮廓信息的表达方式是十分必要的。

1. EHD 描述子

边缘直方图（EHD）描述子是 MPEG-7 标准中提出的一种边缘描述子，它具有描述图像亮度变化的方向和频率的能力。其具体算法原理如下。

（1）对图像分块

当给定一张图像时，不重叠地将其分成 4　4 的子图，且每一子图继续分成若干不重叠的方形子块。在这一过程中，不论图像的大小，均将每一子图划分成数量一定的子块。这样做的目的是去除图像的大小（即分辨率）对特征的影响。通常情况下，每一子图中子块的数量是 2 的倍数，因而在某些情况下可能要舍弃边缘的像素点以满足条件。

(2) 判断每一子块是否包含边缘

当给定图像子块之后，若要判断该子块中是否包含边缘特征，一种简单的办法是对图像在空域利用滤波器进行滤波。在 EHD 特征中，统计以下五种形式的边缘特征：垂直（ o 0 ）、水平（ o 90 ）、o 45 角、o 135 角以及非方向性边缘，如图 1 所示。

（a）垂直　（b）水平　（c）45°角　（d）135°角　（e）非方向

图 1 五种类型边缘

对于某一图像子块，首先将它划分成不重叠的 2×2 的小块，并分别计算每一小块中灰度的平均值，分别记作 a1 a2 a3 a4。同时，可以表示五种滤波器系数 ，

$f_v(k)$、$f_h(k)$、$f_{45}(k)$、$f_{135}(k)$、$f_{non}(k)$，其中 $k \in [1,4]$。

其中。此时，可用这五种滤波器分别计 算图像子块对应的幅值，计算公式如式 1 所示

$$m = \sum_{k=1}^{4} a_k \times f(k) \qquad (1)$$

此时，若某一方向的幅值大于某一阈值，则该子块便被认定为含有该方向的边缘信息，否则该子块不含有边缘信息。

$f_v(0)=1$	$f_v(1)=-1$
$f_v(2)=1$	$f_v(3)=-1$

$f_h(0)=1$	$f_h(1)=1$
$f_h(2)=-1$	$f_h(3)=-1$

$f_{d\text{-}45}(0)=\sqrt{2}$	$f_{d\text{-}45}(1)=0$
$f_{d\text{-}45}(2)=0$	$f_{d\text{-}45}(3)=-\sqrt{2}$

$f_{d\text{-}135}(0)=0$	$f_{d\text{-}135}(1)=\sqrt{2}$
$f_{d\text{-}135}(2)=-\sqrt{2}$	$f_{d\text{-}135}(3)=0$

$f_{nd}(0)=2$	$f_{nd}(1)=-2$
$f_{nd}(2)=-2$	$f_{nd}(3)=2$

（a）垂直　（b）水平　（c）45°角　（d）135°角　（e）非方向

图 2　边缘检测滤波系数

(3)　统计边缘信息

通过以上步骤，可以判断出图像子块中是否包含边缘信息，下面要对这些边缘信息做以统计。在前面提到了在EHD 描述子中将图像划分成4×4 的16 个子图，在这里对每一子图统计5 个方向上各自包含的子块的个数，从而构成一个5 维的向量。接着，对该向量进行归一化，即对向量的每一维均除以该子图中子块的数量。从而，一张图像便可以由一个80 维的向量表示。到这里，EHD 特征的基本生成过程已经完毕。

2. 图像特征的相似性度量

假定为图像库中每幅图像对应的 80 维 EHD 向量， 为查询图像的 EHD 向量，计算向量之间的距离，用于衡量两幅图像是否内容相似。

$$d(f,g)=\sqrt{\sum_{i=1}^{80}\left|f(i)-g(i)\right|^2}$$

按照距离的大小对图像库中的图像进行排序，作为查询图像的检索结果。

大连理工大学实验报告

学院（系）：信通学院 专业：通信工程 班级：电通 1301

姓　　名：温思歆 学号：201383052 组：

实验时间：2016.10.10 实验室：创新园 C219 实验台：

指导教师签字： 成绩：

实验四 基于边缘直方图的图像检索

二、 实验目的和要求

1. 掌握用 matlab 进行图像检索的方法。

二、实验题目、程序和结果

1.完成一个基于 EHD 特征的图像检索系统

主函数 hw4.m 代码：

```
% EHD描述子提取特征向量
clc;clear;
img=imread('A:\working\digital image handle\image\实验三四\Microsoft Imagebase\8.jpg');
loadpath='A:\working\digital image handle\image\实验三四\Microsoft Imagebase\';
format='*.jpg';
fullpath=strcat(loadpath,format);
files=dir(fullpath);

load('database.mat');
sample=hw4_getEHD(img);

for i=1:size(database,3)
    tmp=(sample-database(:,:,i)).*(sample-database(:,:,i));
    similarity(i)=sum(sum(tmp));
end
[~, indices] = sort(similarity);
show_img=img;
[row col ~]=size(img);
for i=1:9
    file=files(indices(i)).name;
    img=imread(strcat(loadpath,file));
    if size(img,3)==1
        img(:,:,2)=img(:,:,1);img(:,:,3)=img(:,:,1);
    end
```

```
    img=imresize(img,[row col]);
    show_img=[show_img img];
end
imshow(show_img);
```

获取 EHD 特征向量函数 hw4_getEHD.m 代码：

```
    function dire=hw4_getEHD(img)
    % EHD描述子提取特征向量
    % clc;clear;
    % img=rgb2gray(im2double(imread('A:\working\digital image handle\image\实验三四
\Microsoft Imagebase\1.jpg')));
    img=(im2double(img));
    if size(img,3)==3
        img=rgb2gray(img);
    end
    %% 步骤1 将图像分为4*4=16个子图
    [row col]=size(img);
    sub_row=round(row/4);
    sub_col=round(col/4);
    img=imresize(img,[sub_row*4 sub_col*4]);
    for i=0:3
        for j=0:3
            sub_graph(:,:,i*4+j+1) =
img(1+sub_row*i:sub_row*(i+1),1+sub_col*j:sub_col*(j+1));
        end
    end
    %% 步骤2 将每个子图分成不重叠的2*2的小块。将每个小块的灰度值计为a1, a2, a3, a4
    cnt_row=4;cnt_col=4;
    subsub_row=floor(sub_row/cnt_row);
    subsub_col=floor(sub_col/cnt_col);
    for k=1:16
        for i=0:3
            for j=0:3
                subsub_graph(:,:,i*4+j+1,k) =
sub_graph(1+subsub_row*i:subsub_row*(i+1),1+subsub_col*j:subsub_col*(j+1),k);
            end
        end
    end
    ave_gray=reshape(mean(mean(subsub_graph)),16,[]);
    %% 步骤3 对于五个方向边缘，定义五种边缘信息
    filt=[1 -1 1 -1;1 1 -1 -1;2^(1/2), 0, 0, -2^(1/2);0,2^(1/2)   , -2^(1/2) , 0;2, -2, -2, 2];
    thre=1;
    %% 步骤4 计算每个小块的每个方向的幅值，其中f分别对应上述五个方向
    for j=1:16
        tmp(:,1)=[ave_gray(1,j) ave_gray(2,j) ave_gray(5,j) ave_gray(6,j)]';
```

```
        tmp(:,2)=[ave_gray(3,j) ave_gray(4,j) ave_gray(7,j) ave_gray(8,j)]';
        tmp(:,3)=[ave_gray(1+8,j) ave_gray(2+8,j) ave_gray(5+8,j) ave_gray(6+8,j)]';
        tmp(:,4)=[ave_gray(3+8,j) ave_gray(4+8,j) ave_gray(7+8,j) ave_gray(8+8,j)]';
        mult_sum=filt*tmp;
        mult_sum(mult_sum>0.2)=1;mult_sum(mult_sum<=0.2)=0;
        dire(j,:)=sum(mult_sum');
    end
```

数据库训练获取特征向量函数 hw4_training.m 代码：

```
%   批量处理得到整个数据集的offset
clc;clear
loadpath='A:\working\digital image handle\image\实验三四\Microsoft Imagebase\';
format='*.jpg';
fullpath=strcat(loadpath,format);
files=dir(fullpath);
for i=1:length(files)
file=files(i).name;
img=imread(strcat(loadpath,file));
database(:,:,i)=hw4_getEHD(img);
end
save('database.mat','database');
```

总体匹配效果较好，但还是存在很多不理想匹配结果，因为数据库图片有限和特征描述算子不够好。

拓展，使用 matlab 自带的 surf 描述算子来做匹配：

主函数 expand.m：

```
clc;clear;
img=im2double(imread('A:\working\digital image handle\image\实验三四\Microsoft
Imagebase\60.jpg'));
raw=im2uint8(img);
if size(img,3)==3
```

```
    img=rgb2gray(img);
end

loadpath='A:\working\digital image handle\image\实验三四\Microsoft Imagebase\';
format='*.jpg';
fullpath=strcat(loadpath,format);
files=dir(fullpath);

load('surf_database.mat');
points=detectSURFFeatures(img);
[sample_descriptor,vpts1] = extractFeatures(img,points);

for i=1:size(database,2)
    database_descriptor=database{i};
    indexPairs = matchFeatures(sample_descriptor,database_descriptor) ;

    similarity(i)=size(indexPairs,1);
end

[~, indices] = sort(similarity);
show_img=raw;
[row col ~]=size(img);
for i=size(indices,2):-1:size(indices,2)-8
file=files(indices(i)).name;
img=imread(strcat(loadpath,file));
if size(img,3)==1
    img(:,:,2)=img(:,:,1);img(:,:,3)=img(:,:,1);
end
img=imresize(img,[row col]);
show_img=[show_img img];
end
imshow(show_img);
```

数据库训练获取特征向量函数expand_training.m代码：

```
clc;clear
loadpath='A:\working\digital image handle\image\实验三四\Microsoft Imagebase\';
format='*.jpg';
fullpath=strcat(loadpath,format);
files=dir(fullpath);

for i=1:length(files)
file=files(i).name;
img=im2double(imread(strcat(loadpath,file)));
if size(img,3)==3
    img=rgb2gray(img);
```

```
end
points=detectSURFFeatures(img);
[descriptor,vpts1] = extractFeatures(img,points);
database{i}=descriptor;
end
save('surf_database.mat','database');
```

总体匹配效果较差，因为数据库图像为 100*100*3，导致特征点较少，且 surf 算子的匹配算法较苛刻，故匹配点数较少，导致效果较差。

三、实验分析和总结

1. 图像检索系统需要提取特征值、训练数据集等，步骤应为先将特征提取函数写好，再训练整个数据集，将数据集的特征向量保存下来，然后再编写匹配算法。
2. 本次实验是提取 EHD 描述算子，依照教程的步骤一步步实现即可，但应该注重细节；在提取边缘信息可用数组的乘法运算，方便快捷；为增加系统的可靠性应该增加一些判断，比如判断输入图像是彩色还是灰度、用 imresize 来保证图像是 4 的整数倍等。
3. 本次实验数据集量较少，故我在网上下载了一些数据集，但是检索效果都不太理想，应该是因为 EHD 描述算子的鲁棒性不好，不可靠。要增强效果，应该采用较好的描述算子，故我利用 matlab 自带的 surf 算子，发现检索效果也不理想，这方便的工作有待研究。