实验一 图像基本操作

一、实验目的

1. 利用 matlab 获取图像
2. 图像直方图均衡
3. 图像的点运算
4. 图像的几何变换

二、实验原理

1. 利用 matlab 获取图像

（a）imread

imread 函数用于读入各种图像文件，其一般的用法为

[X，MAP]=imread(‘filename’, ‘fmt’)

其中，X 为读出的图像数据，MAP 为颜色表数据（或称调色板，亦即颜色索引矩阵，对灰度图像和 RGB 彩色图像，该 MAP 为空矩阵），fmt 为图像的格式（可以缺省），filename 为读取的图像文件（可以加上文件的路径）。

例：[X，MAP]=imread(‘flowers.tif’, ‘tif’)

（b）imwrite

imwrite 函数用于输出图像，其语法格式为：

imwrite(X, MAP, filename, fmt)

imwrite(X, MAP, filename, fmt) 按照 fmt 指定的格式将图像数据矩阵 X 和调色板 MAP 写入文件 filename。

（c）imshow

MATLAB 图像处理工具箱提供了 imshow 函数来显示各种图像，其语法如下：

imshow(I, n)

或 imshow(I_BW)；imshow(X, MAP)；imshow(I_RGB)

其中 imshow(I, n)用于显示灰度图像，I 是图像数据矩阵，n 为灰度级数目（n 可缺省，缺省值为 256）。其它的分别用于显示二值图像、索引色图像和 RGB 真彩色图像。另外，对 RGB 彩色图像，还可以用 imshow(RGB(:, :, 1))、imshow(RGB(:, :, 2))、imshow(RGB(:, :, 3))分别显示 RGB 图像的 R、G、B 三个分量（注意：这样显示出的图像是以各分量值为对应的灰度值所显示的灰度图像）。

需要显示多幅图像时，可以使用 figure 语句，它的功能就是打开一个新的图像显示窗口。也可以使用 subplot 函数将多幅图像显示在同一个图像显示窗口的不同区域位置。

2. 图像直方图均衡

灰度直方图是将数字图像中的所有像素，按照灰度值的大小，统计其所出现的频度。通常，灰度直方图的横坐标表示灰度值，纵坐标为像素个数，也可以采用某一灰度值的像素数占全图像素数的百分比作为纵坐标。

直方图均衡方法的基本原理是：对在图像中像素个数多的灰度值（即对画面起主要作用的灰度值）进行展宽，而对像素个数少的灰度值(即对画面不起主要作用的灰度值)进行归并。从而达到清晰图像的目的。

直方图是灰度级数的函数，反映了图像中具有该灰度级数的像素的个数。直方图均衡化就是把原图像的灰度直方图从比较集中的某个区间变成在全部灰度范围内的均匀分布。均衡化

后图像的像素重新分配，使一定灰度范围内的像素数量大致相同。图像的亮度变得更亮了，增强了图像的整体效果。均衡化后直方图趋于平坦化，灰度间隔（动态范围）拉大，对比度加强，图像清晰，便于读取、分析和处理。

MATLAB 图像处理工具箱中提供的 histeq 函数，可以实现直方图的均衡化。

对于灰度图像，histeq 函数的基本调用格式为

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。

---

3. 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。

图像的线性化处理可以加大图像动态范围，扩展对比度，使图像清晰，特征明显，大大改善人眼的视觉效果。在线性化处理过程中，各个参数要根据新的直方图以及图像效果进行调整，直到满意再最后确定。没有一个公式的参数是固定不变的。

*GST*函数$f(D)$为线性，即

$D_B = f(D_A) = \alpha D_A + b$

显然，

*若$a=1, b=0$,图象像素不发生变化；

*若$a=1, b\neq 0$,图象所有灰度值上移或下移；

*若$a>1$，输出图象对比度增强；

*若$0<a<1$,输出图象对比度减小；

*若$a<0$,暗区域变亮，亮区域变暗，图象求补。

4. 图像的几何变换（图像的缩放，旋转和镜像）

■ **图像的缩放**

MATLAB 图像处理工具箱中的函数 imresize 可以用三种方法对图像进行插值缩放，如果不指定插值方法，则默认为最邻近插值法。

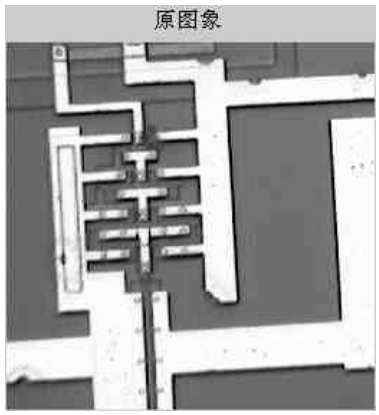
imresize 函数的语法格式为：

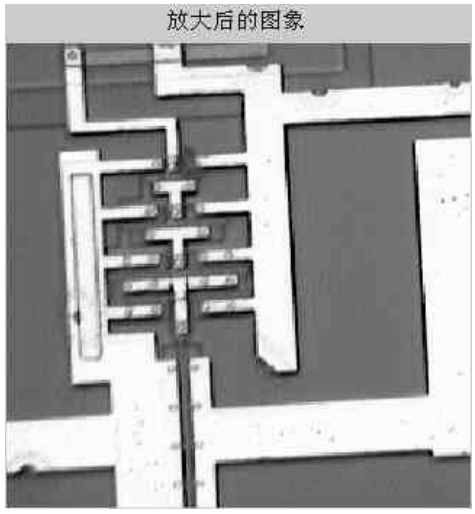
B = imresize(A, m, ‘method’)

上式返回原图像 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'。

例：

```
I = imread('ic.tif');
J = imresize(I, 1.25);
imshow(I), title('原图像')
figure,imshow(J), title('放大后的图像')
```





■ **图像的旋转**

在工具箱中的函数 imrotate 可用上述三种方法对图像进行插值旋转，默认的插值方法也是最邻近插值法。

imrotate 的语法格式为：

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。

例：

```
I = imread('rice.tif');
J = imrotate(I, 30, 'bilinear');
imshow(I); title('原图像')
figure, imshow(J); title('旋转后的图像')
```

三、实验内容

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
3. 图像线性变换和非线性变换

$$D = I + 50$$
$$D = 1.5 \times I$$
$$D = 0.8 \times I$$
$$D = -I + 255$$
$$D = I + \frac{0.8 \times I \times (255 - I)}{255}$$

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。
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。。

实验二 图像的边缘检测

一、实验目的

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、实验原理

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 $\alpha=\tan^{-1}\left[\frac{\partial f}{\partial y}/\frac{\partial f}{\partial x}\right]$，方向导数的最大值是 $|G|=\left[\left(\frac{\partial f}{\partial x}\right)^2+\left(\frac{\partial f}{\partial y}\right)^2\right]^{\frac{1}{2}}$ 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，如：$|\Delta_x f(i,j)|+|\Delta_y f(i,j)|$ 、 $\max(|\Delta_x f(i,j)|,|\Delta_y f(i,j)|)$ 及 $\max|f(i,j)-f(m,n)|$ 等。另外，还有一些常用的算子，如 Roberts 算子和 Sobel 算子。

Roberts 算子的表达式为：

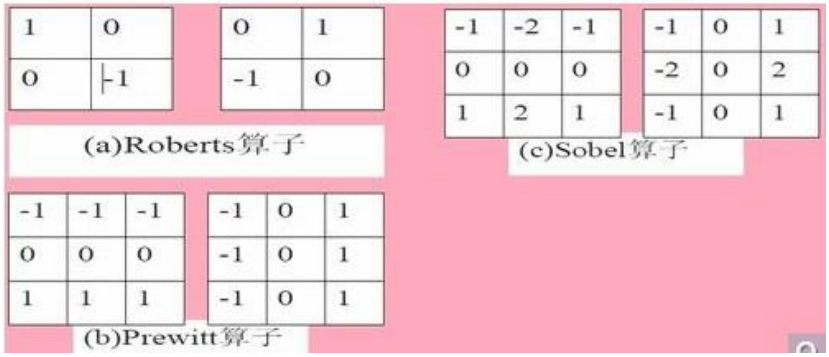
$$\max(|f(i,j)-f(i+1,j+1)|,|f(i+1,j)-f(i,j+1)|)$$

Sobel 算子的表达式为：

X 方向算子：$\begin{pmatrix}1&2&1\\0&0&0\\-1&-2&-1\end{pmatrix}$　　y 方向算子：$\begin{pmatrix}1&0&-1\\2&0&-2\\1&0&-1\end{pmatrix}$

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。

比较经典的梯度算子：



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。

三、实验内容

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。

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。

实验三　图像的空域滤波和频域处理

一、实验目的

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、实验原理

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：

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

(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：

$$M_1=\begin{pmatrix}0 & \frac{1}{5} & 0\\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5}\\ 0 & \frac{1}{5} & 0\end{pmatrix}=\frac{1}{5}\begin{pmatrix}0 & 1 & 0\\ 1 & 1 & 1\\ 0 & 1 & 0\end{pmatrix} \tag{3.3}$$

$$M_2=\begin{pmatrix}\frac{1}{9} & \frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9}\end{pmatrix}=\frac{1}{9}\begin{pmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{pmatrix} \tag{3.4}$$

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。

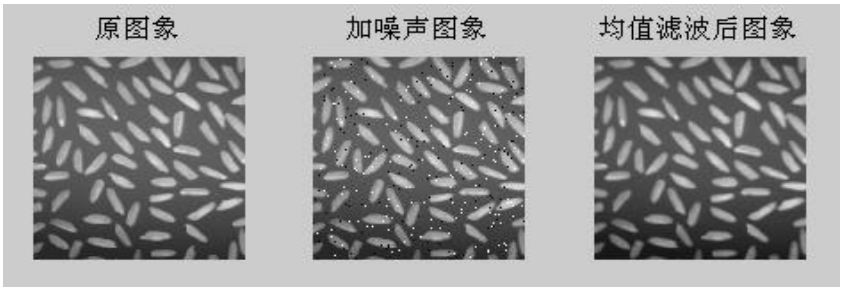
type= 'unsharp'，为对比度增强滤波器，参数 alpha 用于控制滤波器的形状，范围为[0,1]，默认值为 0.2。

下面举一个均值滤波的例子：

```
I=imread('rice.tif');

J=imnoise(I, 'salt & pepper', 0.02);

h=fspecial('average', 3);

I2=filter2(h, J);

subplot(1,3,1), imshow(I), title('原图像');

subplot(1,3,2), imshow(J), title('加噪声图像');

subplot(1,3,3), imshow(I2, [ ]), title('均值滤波后图像');
```

（注意"imshow(I2, [ ])"中的参数"[ ]"！它表示由程序自动调整图像数据的类型与范围，以便正确显示图像）

程序执行的结果如图：



在 MATLAB 图像处理工具箱中，提供了 medfilt2 函数用于实现二维中值滤波。

Medfilt2 函数的语法格式为：

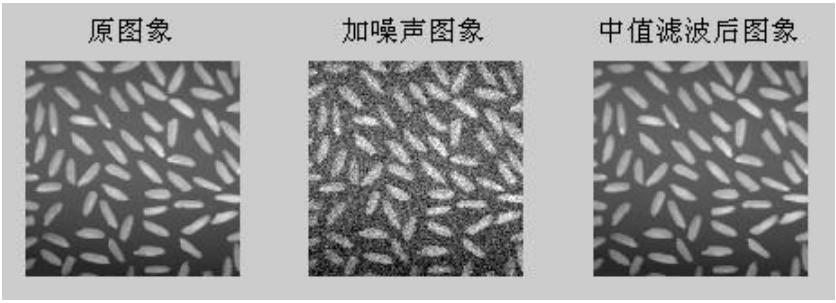
B = medfilt2(A)　%用 3×3 的滤波窗口对图像 A 进行中值滤波。

B = medfilt2(A,[m n]) %用指定大小为 m×n 的窗口对图像 A 进行中值滤波。

以下举例说明：

```
I=imread('rice.tif');
J=imnoise(I, 'gaussian', 0.02);
I2=medfilt2(J, [3, 3]);
subplot(1,3,1), imshow(I), title('原图像');
subplot(1,3,2), imshow(J), title('加噪声图像');
subplot(1,3,3), imshow(I2), title('中值滤波后图像');
```

运行的结果如图；



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：

$$F(p,q)=\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}f(m,n)e^{-j(2\pi/M)pm}e^{-j(2\pi/N)qn} \qquad \text{p=0,1…M-1}$$

q=0,1…N-1

(6.1)

或

$$F(\omega_1,\omega_2)=\sum_{m=0}^{M-1}\sum_{n=1}^{N-1}f(m,n)e^{-j\omega_1 m}e^{-j\omega_2 n} \quad \text{p=0,1…M-1} \quad \text{q=0,1…N-1}$$

（6.2）

离散傅立叶反变换的定义如下：

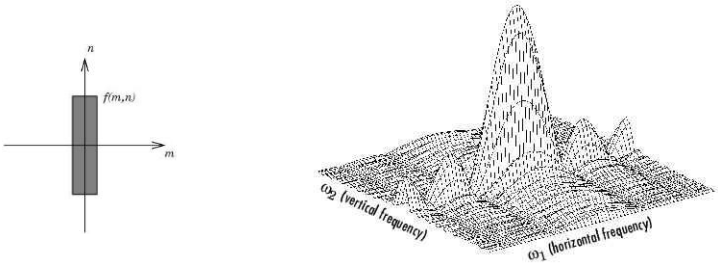
$$f(m,n)=\frac{1}{MN}\sum_{p=0}^{M-1}\sum_{q=0}^{N-1}F(p,q)e^{j(2\pi/M)pm}e^{j(2\pi/N)qn}$$

m=0,1…M-1　n=0,1…N-1

(6.3)

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（$w1$，$w2$）。

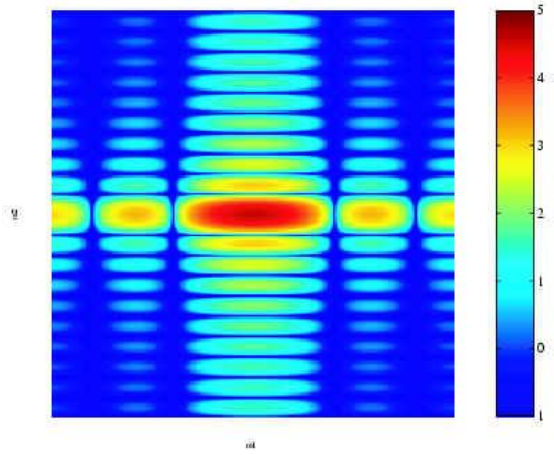
例如，函数f（m, n）在一个矩形区域内函数值为 1，而在



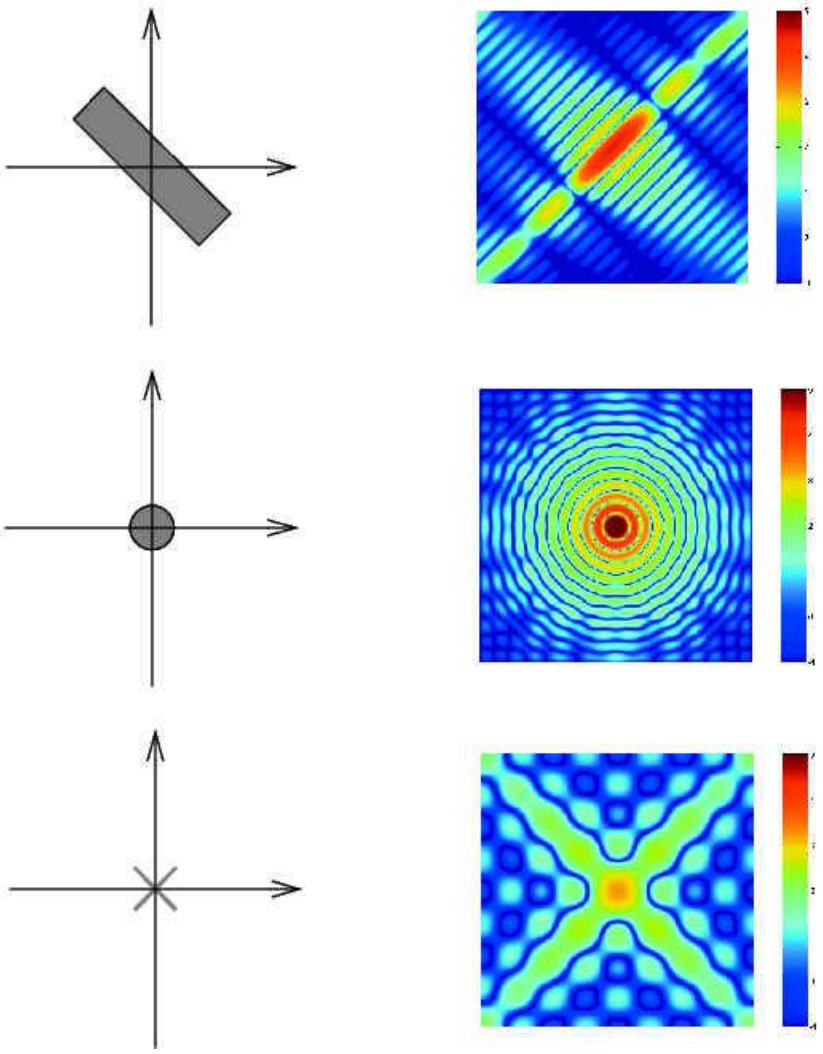
其他区域为 0，如图所示。

为了简便起见，假设f（m, n）为一个连续函数，则f（m, n）的傅立叶变换的幅度值（即$|F(\omega_1,\omega_2)|$）显示为网格图，如图所示。

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$\log|F(\omega_1,\omega_2)|$，如图所示。



几种简单函数的傅立叶变换的频谱可以直观的表示为图所示的样子。



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。

三、实验内容

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。

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。

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。

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。

实验四 基于边缘直方图的图像检索

一、实验目的

1. 掌握用 matlab 进行图像检索的方法。

二、实验原理

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（0^o）、水

平（90^o）、45^o 角、135^o 角以及非方向性边缘，如图 1 所示。

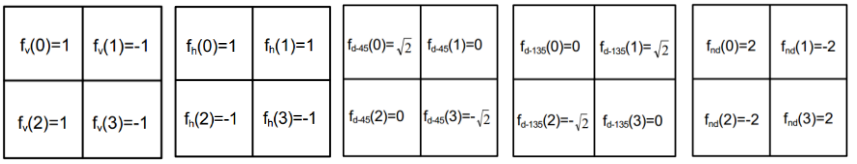
（a）垂直 （b）水平 （c）45^o 角（d）135^o 角 （e）非方向

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 a_1、a_2、a_3、a_4。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 $k \in [1,4]$。此时，可用这五种滤波器分别计算图像子块对应的幅值 m，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。



（a）垂直 （b）水平 （c）45^o 角 （d）135^o 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 f 为图像库中每幅图像对应的 80 维 EHD 向量，g 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。

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

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。

三、实验内容

完成一个基于 EHD 特征的图像检索系统。

实验五 图像的增强

一、实验目的

实现图像的平滑（去噪声），中值滤波，锐化等处理。

二、实验原理

1．平滑(smoothing)

图像的平滑用信号处理的理论来解释，这种做法实现的是一种简单的低通滤波器(low pass filter)。在灰度连续变化的图像中，如果出现了与相邻象素的灰度相差很大的点，这种情况被认为是一种噪声。灰度突变在频域中代表了一种高频分量，低通滤波器的作用就是滤掉高频分量，从而达到减少图像噪声的目的。平滑模板的思想是通过将一点和周围 8 个点作平均，从而去除突然变化的点，滤掉噪声，其代价是图像有一定程度的模糊。这种做法实现起来很简单。每一点的灰度和它周围八个点的灰度相加，然后除以 9，作为新图中对应点的灰度，数学模型可以表示如下：

$$\frac{1}{9}\times\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

这种表示方法有点象矩阵，我们称其为模板(template)。中间的点表示中心元素，即，用哪个元素做为处理后的元素。这是一种平滑模板，称之为 Box 模板。

Box 模板虽然考虑了邻域点的作用，但并没有考虑各点位置的影响，对于所有的 9 个点都一视同仁，所以平滑的效果并不理想。实际上我们可以想象，离某点越近的点对该点的影响应该越大，为此，我们引入了加权系数，距离越近的点，加权

系数越大。数学模型可以表示如下：$\frac{1}{16}\times\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$

系数加权后的新模板也是一个常用的平滑模板，称为高斯(Gauss)模板。它是通过采样 2 维高斯函数得到的。

2．中值滤波

中值滤波也是一种典型的低通滤波器，它的目的是保护图像边缘的同时去除噪声。所谓中值滤波，是指把以某点(x,y)为中心的小窗口内的所有象素的灰度按从大到小的顺序排列，将中间值作为(x,y)处的灰度值(若窗口中有偶数个象素，则取两个中间值的平均)。

中值滤波的特点是容易去除孤立点，线的噪声同时保持图像的边缘；它能很好的去除二值噪声，但对高斯噪声无能为力。要注意的是，当窗口内噪声点的个数大于窗口宽度的一半时，中值滤波的效果不好。这是很显然的。

3．锐化(sharpening)

锐化和平滑恰恰相反，它是通过增强高频分量来减少图像中的模糊，因此又称为高通滤波(high pass filter)。锐化处理在增强图像边缘的同时增加了图像的噪声。常用的锐化模板是拉普拉斯(Laplacian)模板如下，又是个数学家的名字，可见学好数学，走遍天下都不怕。

$$\begin{bmatrix}-1 & -1 & -1\\ -1 & 9 & -1\\ -1 & -1 & -1\end{bmatrix}$$

拉普拉斯模板的作法：先将自身与周围的 8 个象素相减，表示自身与周围象素的差别；再将这个差别加上自身作为新象素的灰度。可见，如果一片暗区出现了一个亮点，那么锐化处理的

结果是这个亮点变得更亮，增加了图像的噪声。因为图像中的边缘就是那些灰度发生跳变的区域，所以锐化模板在边缘检测中很有用。

要注意的是，运算后如果出现了大于 255 或者小于 0 的点，称为溢出，溢出点的处理通常是截断，即大于 255 时，令其等于 255；小于 0 时，取其绝对值。

4．彩色编码

对于一幅灰度图像，我们需要将它人为地转换成一幅彩色图像，这在医学图像处理中是很常见的。因为人眼对灰度微弱递变的敏感程度远远小于对彩色变化的敏感程度。因此，将一幅灰度图像按照特定的彩色编码进行彩色变换，这样就可以看到图像更加精细的结构，以便于医师对疾病的诊断。一般，要将灰度图进行为彩色变换，可以采用一个 256 色的调色板（我们称之为为彩色编码表），在里面我们定义了每个灰度对应的颜色的 RGB 值。

5．Pavlidis 经典细化算法

对二值图像 ，用 Pavlidis 的经典细化算法细化黑区域（象素值为 1）。Pavlidis 的经典细化算法是典型的并行算法，通过与两个模板进行匹配层剥边界象素。所得到的骨架形态是 8一连接的。处理的过程是：先令要处理的像素值为-1，然后用模板去匹配，先用第一个模板匹配，如果匹配。记录下来该点设它的值为 1。如果不匹配，再用第二个模板匹配，同样记录下来匹配的点令它的值为 1。依次到第六个模板记录下匹配的像素，这些像素就是骨架像素。而值为-1 即为将要清除的像素。

三、实验内容

1、对实验三图像 1，利用 imnoise 函数在图像中加入高斯噪声（噪声的均值和方差任意）、椒盐噪声（噪声密度自定）

2、对题目 1 加入噪声后的图像，利用邻域平均法对其进行平滑去噪，观察去噪后的图像

3、对题目 1 中加入噪声后的图像进行中值滤波,观察滤波效果，并与 2 中的结果进行对比。

fspecial：设计一个指定类型的滤波器

filter2：用指定的滤波器对图像进行滤波

实验六 图像的分析

一、实验目的

介绍实现图像分析的各种方法。

二、实验原理

图像分割有三种不同的途径，其一是将各象素划归到相应物体或区域的象素聚类方法即区域法，其二是通过直接确定区域间的边界来实现分割的边界方法，其三是首先检测边缘象素再将边缘象素连接起来构成边界形成分割。 图像分割是图像理解的基础，而在理论上图像分割又依赖图像理解，彼此是紧密关联的。图像分割在一般意义下是十分困难的问题，目前的图像分割一般作为图像的前期处理阶段，是针对分割对象的技术，是与问题相关的，如最常用到的利用阈值化处理进行的图像分割

1．阈值分割

阈值是在分割时作为区分物体与背景象素的门限，大于或等于阈值的象素属于物体，而其它属于背景。这种方法对于在物体与背景之间存在明显差别（对比）的景物分割十分有效。实际上，在任何实际应用的图像处理系统中，都要用到阈值化技术。为了有效地分割物体与背景，人们发展了各种各样的阈值处理技术，包括全局阈值、自适应阈值、最佳阈值等等。 全局阈值

全局阈值

是指整幅图像使用同一个阈值做分割处理。适用于背景和前景有明显对比的图像。

自适应阈值

在许多情况下，物体和背景的对比度在图像中不是各处一样的，这时很难用统一的一个阈值将物体与背景分开。这时可以根据图像的局部特征分别采用不同的阈值进行分割。实际处理时，需要按照具体问题将图像分成若干子区域分别选择阈值，或者动态地根据一定的邻域范围选择每点处的阈值，进行图像分割。

最佳阈值

阈值的选择需要根据具体问题来确定，一般通过实验来确定。对于给定的图像，可以通过分析直方图的方法确定最佳的阈值，例如当直方图明显呈现双峰情况时，可以选择两个峰值的中点作为最佳阈值。

2．差影检测

差影法的原理非常简单：将前后两幅图像相减，得到的差作为结果图像。差影法是非常有用的，比如说可以用在监控系统中。在银行金库内，摄像头每隔一小段时间，拍摄一幅图，与上一幅图做差影；如果差别超过了预先设置的阈值，说明有人，这时就应该拉响警报。

图像的分割与检测(识别)实际上是一项非常困难的工作。由于人类在观察图像时适用了大量的知识，所以没有任何一台计算机在分割和检测真实图像时，能达到人类视觉系统的水平。正因为如此，对于大部分图像应用来说，自动分割与检测还是一个将来时。目前只有少数的几个领域(如印刷体识别 OCR)自动识别达到了实用的水平。目前，我们能从一幅图像中获得的信息只是每个象素的颜色或灰度值，除此以外别无其它，完成上述功能实在是太困难了。所以说解决图像分割和检测最根本的方法是在编码(成象)时就给予考虑。这也正是 MPEG4 及未来的视频压缩编码标准的主要工作。正因为有上述的困难，所以我们要介绍的只是一些最基本，最简单的算法和思想，只能是一些具体(而不是通用)的应用。

三、实验内容

选择一幅图像，利用阈值分割法对一幅灰度图像进行灰度分割处理：

1）选择一个初始阈值 T，建议初始阈值选择为图像中最大亮度和最小亮度的中间值

2）使用 T 分割图像，这样会产生两组像素，亮度值>=T 的所有像素组成 G1，亮度值<T 的所有像素组成 G2；

3）计算 G1 和 G2 范围内的像素的平均亮度 U1 和 U2

4）计算一个新的阈值 T=1/2×（U1+U2）

5）重复 2）～4）步骤，知道连续迭代中 T 的差比预先指定的参数 T0 小为止。

6）利用最后确定的阈值 T，对图像进行二值化分割。

实验七 图像的编码

一、实验目的

介绍实现图像编码的各种方法。

二、实验原理

1．LZW 编码

LZW 是一种比较复杂的压缩算法，其压缩效率也比较高。我们在这里只介绍一下它的基本原理：LZW 把每一个第一次出现的字符串用一个数值来编码，在还原程序中再将这个数值还成原来的字符串。例如：用数值 0x100 代替字符串“abccddeee”，每当出现该字符串时，都用 0x100 代替，这样就起到了压缩的作用。至于 0x100 与字符串的对应关系则是在压缩过程中动态生成的，而且这种对应关系隐含在压缩数据中，随着解压缩的进行这张编码表会从压缩数据中逐步得到恢复，后面的压缩数据再根据前面数据产生的对应关系产生更多的对应关系，直到压缩文件结束为止。LZW 是无损的。GIF 文件采用了这种压缩算法。 要注意的是，LZW 算法由 Unisys 公司在美国申请了专利，要使用它首先要获得该公司的认可。

哈夫曼编码表

2．哈夫曼(Huffman)

编码是一种常用的压缩编码方法，是 Huffman 于 1952 年为压缩文本文件建立的。它的基本原理是频繁使用的数据用较短的代码代替，较少使用的数据用较长的代码代替，每个数据的代码各不相同。这些代码都是二进制码，且码的长度是可变的。

举个例子：假设一个文件中出现了 8 种符号

S0,S1,S2,S3,S4,S5,S6,S7，那么每种符号要编码，至少需要 3 比特。假设编码成 000,001,010,011,100,101,110,111(称做码字)。那么符号序列　S0S1S7S0S1S6S2S2S3S4S5S0S0S1 编码后变成

000001111000001110010010011100101000000001

共用了 42 比特，我们发现 S0，S1，S2 这三个符号出现的频率比较大，其它符号出现的频率比较小，如果我们采用一种编码方案使得 S0，S1，S2 的码字短，其它符号的码字长，这样就能够减少占用的比特数。

例如，我们采用这样的编码方案：

S0 到 S7 的码字分别：

01,11,101,0000,0001,0010,0011,100，

上述符号序列变成：

011110001110011101101000000010010010111，

共用了 39 比特，尽管有些码字如 S3，S4，S5，S6 变长了(由 3 位变成 4 位)，但使用频繁的几个码字如 S0，S1 变短了，所以实现了压缩。一般进行 Huffman 编码的步骤如下：

(1) 首先统计出每个符号出现的频率，上例 S0 到 S7 的出现频率分别为 4/14，3/14，2/14，1/14，1/14，1/14，1/14，1/14。

(2) 从左到右把上述频率按从小到大的顺序排列。

(3) 每一次选出最小的两个值，作为二叉树的两个叶子节点，将和作为它们的根节点，这两个叶子节点不再参与比较，新的根节点参与比较。

(4) 重复(3)，直到最后得到和为 1 的根节点。

(5) 将形成的二叉树的左节点标 0，右节点标 1。把从最上面的根节点到最下面的叶子节点途中遇到的0,1序列串起来，就得到了各个符号的编码。

产生 Huffman 编码需要对原始数据扫描两遍。第一遍扫描要精确地统计出原始数据中，每个值出现的频率，第二遍是建立 Huffman 树并进行编码。由于需要建立二叉树并遍历二叉树生成编码，因此数据压缩和还原速度都较慢，但简单有效，因而得到广泛的应用。

3．香农弗诺编码

香农弗诺编码也是一种常见的变长编码，利用改变吗由是效率可一高达 100%。一般进行香农弗诺编码的步骤如下：

（1） 首先统计出每个灰度出现的频率

（2）从左到右吧上述频率从小到大的顺序排列

（3）从序列中的某个位置将序列分为两个子序列，并尽量是两个序列频率和近似相等，给前面一个子序列赋值为 1，给后面一个子序列赋值为 0。

（4） 重复步骤（3），直到个个子序列不能再分

（5） 分配码字。将给每个元素所属序列的值串起来，这样就得到了每个元素的香农弗诺编码。

4．行程编码

行程编码(Run Length Coding)的原理也很简单：将一行中颜色值相同的相邻象素用一个计数值和该颜色值来代替。例如 aaabccccccddeee 可以表示为 3a1b6c2d3e。如果一幅图像是由很多块颜色相同的大面积区域组成，那么采用行程编码的压缩效率是惊人的。然而，该算法也导致了一个致命弱点，如果图像中每两个相邻点的颜色都不同，用这种算法不但不能压缩，反而数据量增加一倍。所以现在单纯采用行程编码的压缩算法

用得并不多，PCX 文件算是其中的一种。PCX 文件最早是 PC Paintbrush 软件所采用的一种文件格式，由于压缩比不高，现在用的并不是很多了。它也是由头信息、调色板、实际的图像数据三个部分组成。其中头信息的结构为：其中值得注意的是以下几个数据：manufacturer 为 PCX 文件的标识，必须为 0x0a；xmin 为最小的 x 坐标，xmax 最大的 x 坐标，所以图像的宽度为 xmax-xmin+1，同样图像的高度为 ymax-yin+1；bytes_per_line 为每个编码行所占的字节数，下面将详细介绍。PCX 的调色板在文件的最后。以 256 色 PCX 文件为例，倒数第 769 个字节为颜色数的标识，256 时该字节必须为 12，剩下的 768(256×3)为调色板的 RGB 值。为了叙述方便，我们针对 256 色 PCX 文件，介绍一下它的解码过程。编码是解码的逆过程，有兴趣的读者可以试着自己来完成。

解码是以行为单位的，该行所占的字节数由 bytes_per_line 给定。为此，我们开一个大小为 bytes_per_line 的解码缓冲区。一开始，将缓冲区的所有内容清零。从文件中读出一个字节 C，若 C>0xc0，说明是行程(Run Length)信息，即 C 的低 6 位表示后面连续的字节个数(所以最多 63 个连续颜色相同的象素，若还有颜色相同的象素，将在下一个行程处理)，文件的下一个字节就是实际的图像数据 (即该颜色在调色板中的索引值) 。若 C<0xc0，则表示 C 是实际的图像数据。如此反复，直到这 bytes_per_line 个字节处理完，这一行的解码完成。PCX 就是有若干个这样的解码行组成。

三、实验内容

实现基本 JPEG 的压缩和编码：

1. 首先通过 DCT 变换去除数据冗余；
2. 使用量化表对 DCT 系数进行量化；
3. 对量化后的系数进行 Huffman 编码。