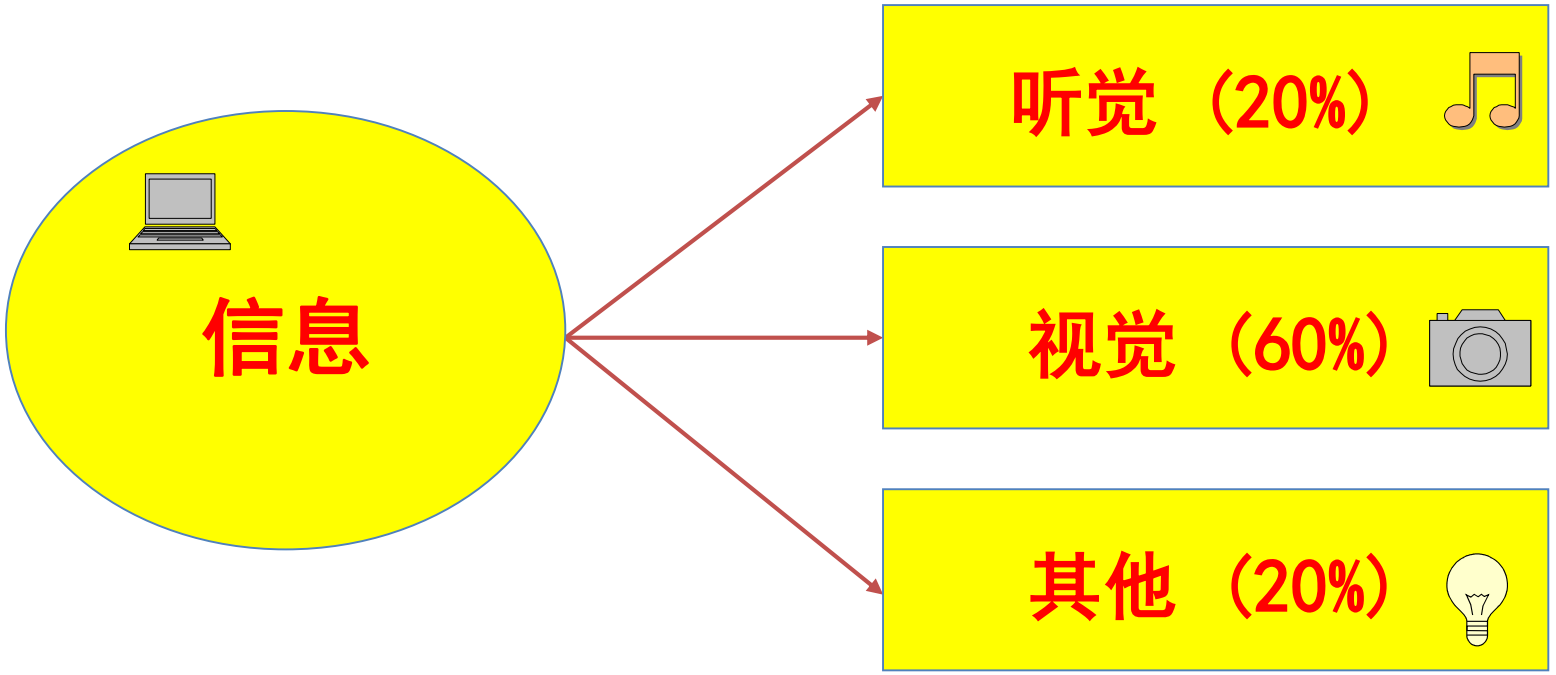
4.3 图像处理工具箱

在MATLAB的图像处理工具箱(Image Processing Toolbox)提供了大量的用于图像处理的函数。我们将对图像及图像处理的相关知识进行介绍，然后介绍matlab中常用的图像操作进行介绍。

4.3.1 图像与数字图像简介

- 图像：利用各种系统观测客观世界获得的且可以直接或间接感知的视觉实体。



- **图像：**图像是对客观世界的反映。“图”是指物体透射光或反射光的分布，“像”是人的视觉对“图”的认识。“图像”是两者的结合。图像既是一种光的分布，也包含人的视觉心理因素。图像的最初取得是通过对物体和背景的“摄取”。这里指的“摄取”即意味着一种“记录”过程，如照相、摄像、扫描等。

- **数字图像：**一幅图片可以定义为一个二维函数f(x, y)，其中x和y表示空间坐标，而f对于任何一对(x, y)坐标的函数值称为该点处图像的亮度或灰度。当x, y和f的值都是有限的、离散的数值时，称该图像为数字图像。

图像:

（1）模拟图像：光学图像、模拟电视图像等。处理速度快，但精度和灵活度差。

（2）数字图像：数码相机、数字电视等。是将连续的模拟图像经过离散化处理后得到的计算机可以识别及处理的点阵图像。

数字图像的优点：

（1）精度高：目前计算机可将模拟图像转化成高精度数字图像

（2）处理方便：数字图像是一组数据，可利用计算机对其处理

（3）重复性好：数字图像可比模拟图像有正常的保质时间

数字图像处理的研究内容

1）图像变换：由于图像阵列很大，直接在空间域中进行处理，涉及计算量很大。因此，往往采用各种图像变换的方法，如傅立叶变换、沃尔什变换、离散余弦变换等间接处理技术，将空间域的处理转换为变换域处理，不仅可减少计算量，而且可获得更有效的处理（如傅立叶变换可在频域中进行数字滤波处理）。目前新兴研究的小波变换在时域和频域中都具有良好的局部化特性，它在图像处理中也有着广泛而有效的应用。

2) 图像编码与压缩：图像编码压缩技术可减少描述图像的数据量（即比特数），以便节省图像传输、处理时间和减少所占用的存储器容量。

压缩可以在不失真的前提下获得，也可以在允许的失真条件下进行。编码是压缩技术中最重要的方法，它在图像处理技术中是发展最早且比较成熟的技术。

3) 图像增强和复原：图像增强和复原的目的是为了提高图像的质量，如去除噪声，提高图像的清晰度等。

图像增强不考虑图像降质的原因，突出图像中所感兴趣的部分。如强化图像高频分量，可使图像中物体轮廓清晰，细节明显；如强化低频分量可减少图像中噪声影响。

图像复原要求对图像降质的原因有一定的了解，一般讲应根据降质过程建立“降质模型”，再采用某种滤波方法，恢复或重建原来的图像。

4) 图像分割：图像分割是数字图像处理中的关键技术之一。图像分割是将图像中有意义的特征部分提取出来，其有意义的特征有图像中的边缘、区域等，这是进一步进行图像识别、分析和理解的基础。虽然目前已研究出不少边缘提取、区域分割的方法，但还没有一种普遍适用于各种图像的有效方法。因此，对图像分割的研究还在不断深入之中，是目前图像处理中研究的热点之一。

5) 图像描述：图像描述是图像识别和理解的必要前提。作为最简单的二值图像可采用其几何特性描述物体的特性，一般图像的描述方法采用二维形状描述，它有边界描述和区域描述两类方法。对于特殊的纹理图像可采用二维纹理特征描述。随着图像处理研究的深入发展，已经开始进行三维物体描述的研究，提出了体积描述、表面描述、广义圆柱体描述等方法。

6) 图像分类（识别）：图像分类（识别）属于模式识别的范畴，其主要内容是图像经过某些预处理（增强、复原、压缩）后，进行图像分割和特征提取，从而进行判决分类。

图像分类常采用经典的模式识别方法，有统计模式分类和句法（结构）模式分类，近年来新发展起来的模糊模式识别和人工神经网络模式分类在图像识别中也越来越受到重视。

数字图像处理研究目的

一般来讲，对图像进行处理（或加工、分析）的主要目的有3个：

1） **提高图像的视感质量**，如进行图像的亮度、彩色变换，增强、抑制某些成分，对图像进行几何变换等，以改善图像的质量。

2） **提取图像中所包含的某些特征或特殊信息**。这些被提取的特征或信息往往为计算机分析图像提供便利。提取特征或信息的过程是模式识别或计算机视觉的预处理。提取的特征可以包括很多方面，如频域特征、灰度或颜色特征、边界特征、区域特征、纹理特征、形状特征、拓扑特征和关系结构等。

3） **图像数据的变换、编码和压缩**，以便于图像的存储和传输。

数字图像处理的应用：

计算机科学领域：进行计算机辅助设计、人工智能研究和多媒体计算机研究；

通信领域：进行图像压缩、图像编码，例如可视电话、卫星通信等

生物医学工程领域：进行医学图像增强和分析等，例如CT图像，超声图像、核磁共振成像图像分析等

。。。。。。

图像数字化过程：扫描、采样、量化

扫描：按照一定的先后顺序对图像进行遍历的过程；

采样：将空间上连续的图像变成离散点的操作，可看做将平面划分成网格的过程；

量化：将采样得到的值转化为离散的整数值。

Matlab支持的文件格式：BMP、GIF、HDP、JPEG、PCX、PNG、TIFF、XWD、CUR、ICO等

4.3.2 图像处理工具箱所支持的图像类型

图像处理工具箱支持4种图像类型，它们是：

- 真彩色图像（RGB images）
- 索引色图像（index images）
- 灰度图像（intensity images）
- 二值图像（binary images）

此外，Matlab还支持由多帧图像组成的图像序列。

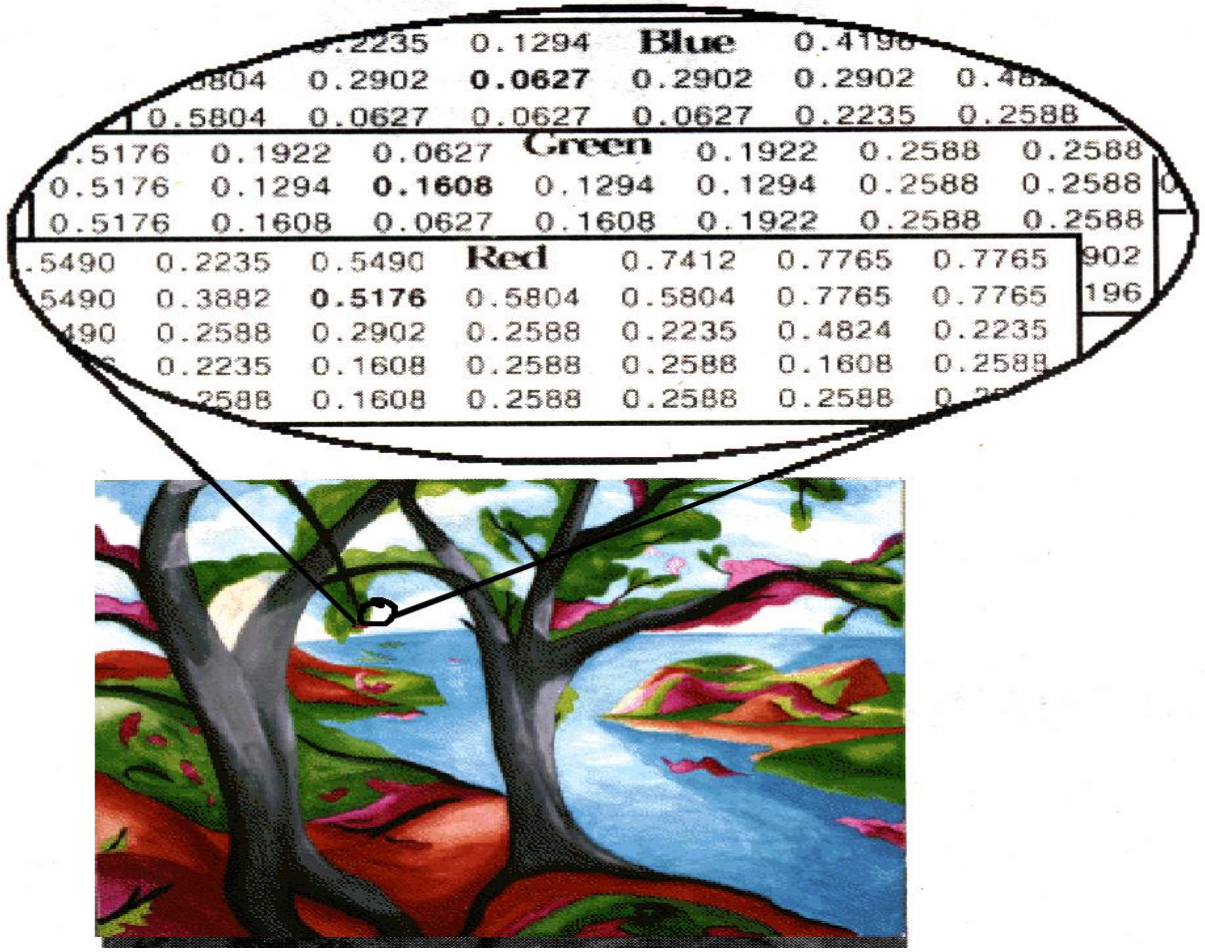
(1) 真彩色图像

真彩色图像用 R、G、B 3个分量表示1个像素的颜色，所以对1个尺寸为m×n的真彩色图像来说，其数据结构就是一个m×n×3的多维数组。如果要读取图像中(100，50)处的像素值，可以查看三元组(100, 50, 1：3)。

真彩色图像可用双精度存储，此时亮度值的范围是［0, 1］。也常用无符号整型存储，亮度值的范围为［0, 255］。

真彩色图像数据格式

双精度类: Double (每个像素占8个字节)	整数类：Uint8 (每个像素占1个字节)
数组大小：m×n×3 (: , : , 1)—红色分量 (: , : , 2)—绿色分量 (: , : , 3)—蓝色分量 像素取值：[0 , 1]	数组大小：m×n×3 (: , : , 1)—红色分量 (: , : , 2)—绿色分量 (: , : , 3)—蓝色分量 像素取值：[0 , 255]


0.2235 0.1294 Blue 0.4196
0.5804 0.2902 0.0627 0.2902 0.2902
0.5804 0.0627 0.0627 0.0627 0.2235 0.2588
0.5176 0.1922 0.0627 Green 0.1922 0.2588 0.2588
0.5176 0.1294 0.1608 0.1294 0.1294 0.2588 0.2588
0.5176 0.1608 0.0627 0.1608 0.1922 0.2588 0.2588
0.5490 0.2235 0.5490 Red 0.7412 0.7765 0.7765
0.5490 0.3882 0.5176 0.5804 0.5804 0.7765 0.7765
0.2588 0.2902 0.2588 0.2235 0.4824 0.2235
0.2235 0.1608 0.2588 0.2588 0.1608 0.2588
0.1608 0.2588 0.2588 0.2588

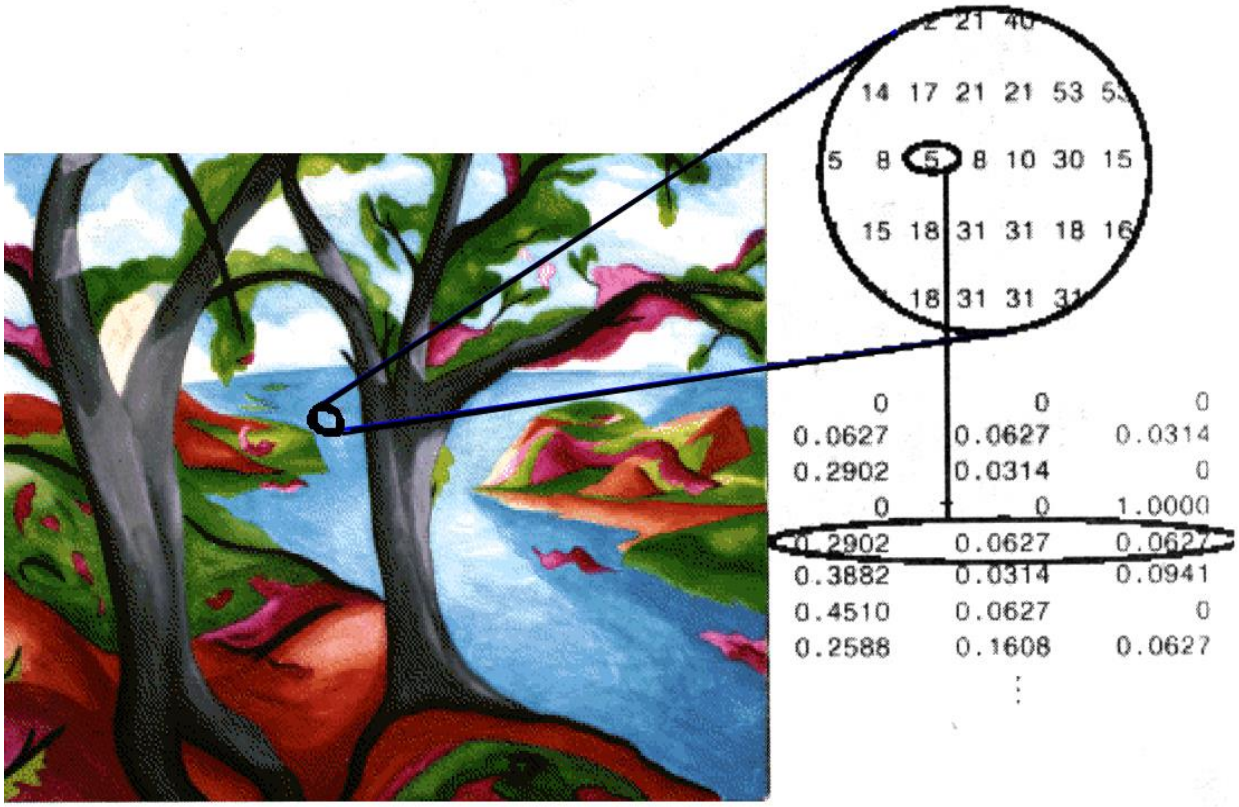
真彩色图像的结构

(2) 索引色图像

索引图像是把像素值直接作为RGB调色板下标的图像。

Matlab中的索引色图像包含2个结构，一个是调色板map；另外一个是图像数据矩阵X。调色板是一个有3列和若干行的色彩映像矩阵，矩阵的每行都代表一种色彩，通过3个分别代表红、绿、蓝颜色强度的双精度数，形成一种特定的颜色。图像数据可以是uint8或是双精度的。

需要注意的是Matlab中的调色板的色彩强度是［0，1］中的浮点数，0代表最暗，1代表最亮。


2 21 40
14 17 21 21 53 53
5 8 5 8 10 30 15
15 18 31 31 18 16
18 31 31 31
0 0 0
0.0627 0.0627 0.0314
0.2902 0.0314 0
0 0 1.0000
0.2902 0.0627 0.0627
0.3882 0.0314 0.0941
0.4510 0.0627 0
0.2588 0.1608 0.0627

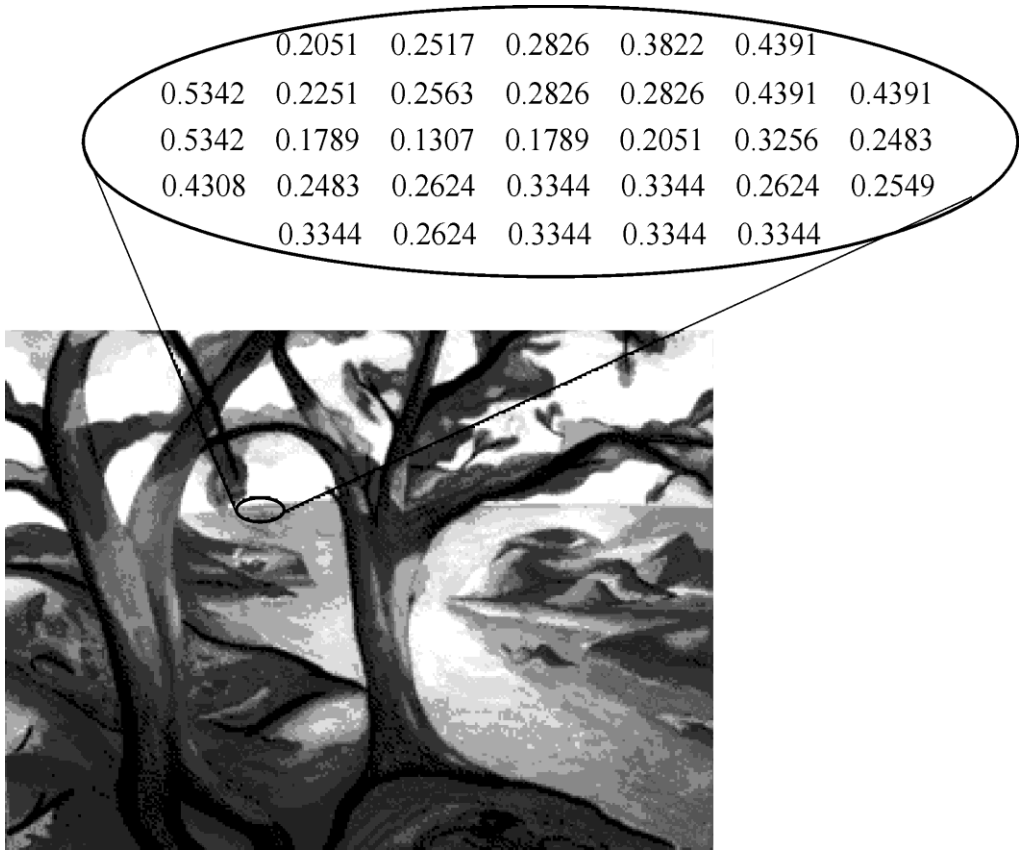
索引色图像的结构

索引图像的数据格式

双精度类:Double (每个元素占8个字节)	整数类：Uint8 (每个元素占1个字节)
图像数组大小：m×n 图像元素取值:[1 , p] 调色板矩阵：p×3	图像数组大小：m×n 图像元素取值:[0 , p-1] 调色板矩阵：p×3

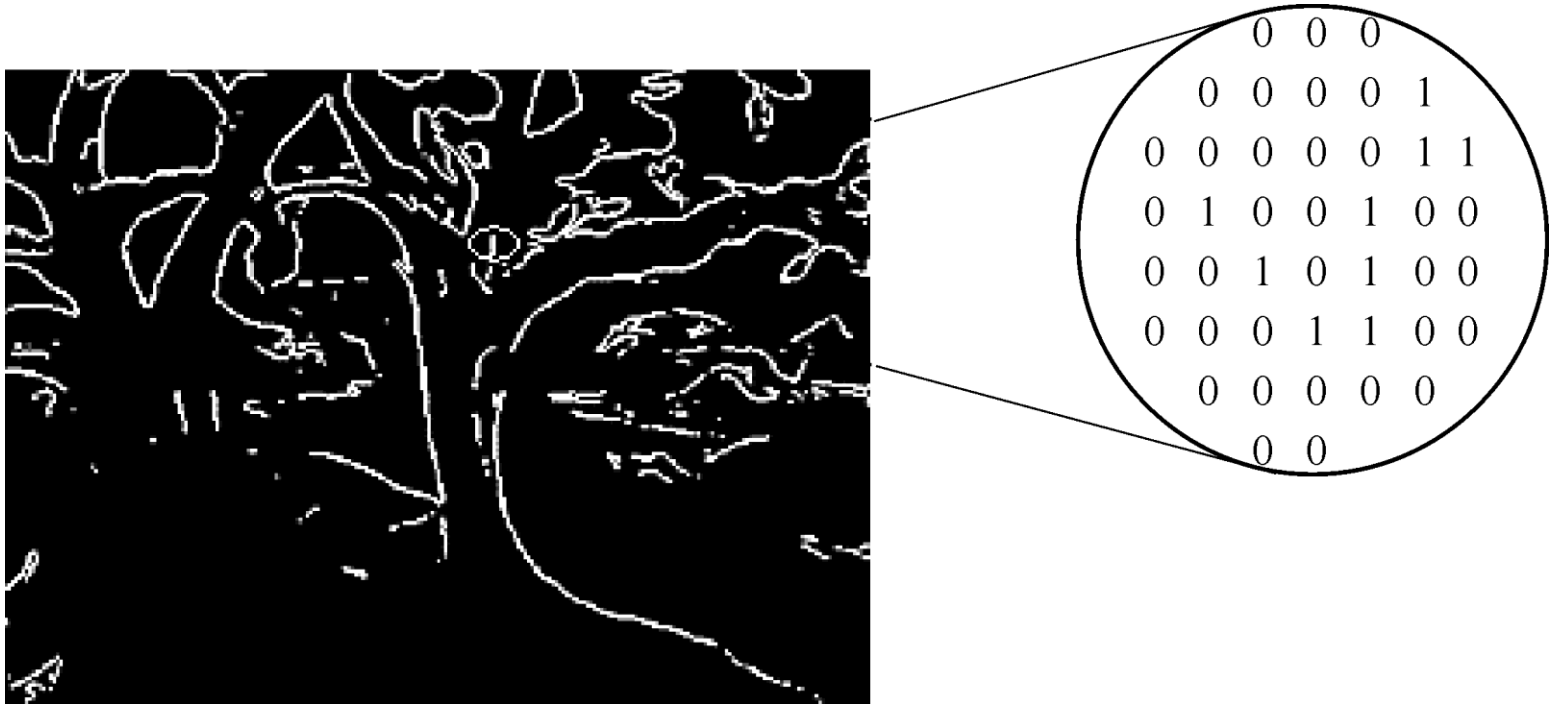
(3) 灰度图像

存储灰度图像只需要一个数据矩阵，数据类型可以是**double**也可以是**uint8**。



(4) 二值图像

与灰度图像相同，二值图像只需一个数据矩阵，每个像素只有2个灰度值。可以采用uint8或double类型存储，工具箱中以二值图像作为返回结果的函数都使用uint8类型。



(5) 图像序列

图像处理工具箱支持将多帧图像连接成图像序列。可以使用Matlab的cat函数将分散的图像合并成图像序列，前提是各图像的尺寸必须相同，如果是索引色图像，调色板也必须是一样的。

比如要将A1、A2、A3、A4、A5五幅图像合并成一个图像序列A，Matlab语句为

A=cat (1，A1，A2，A3，A4，A5)，1维，纵向排列

A=cat (2，A1，A2，A3，A4，A5)，2维，横向排列

A=cat (3，A1，A2，A3，A4，A5)，3维，竖向排列

图像序列也可以产生一个四维的数组，图像帧的序号在图像的长、宽、颜色深度之后构成第四维。

一个包含了5幅400×300真彩色图像的序列，其大小为

400×300×3×5

一个包含了5幅400×300灰度或是索引图像的序列，其大小为

400×300×1×5

A=cat (4，A1，A2，A3，A4，A5)， 4维

提取其中一帧（如第2帧）

A(: , : , : , 2)

4.3.3 图像文件的读写和查询

imread：读取图形文件格式的图像；

imwrite：写入图形文件格式的图像；

imfinfo：获取图像的信息；

load \ save：以Mat文件加载或保存矩阵数据；

imshow：显示加载到Matlab中的图像。 ■

(1) 图形图像文件的读取

利用函数**imread**可以完成图形图像文件的读取操作，其语法如下, ■

一般：A=imread(filename, fmt) ■

索引图像：［X, map］=imread(filename, fmt) ■

imread函数可以从任何Matlab 支持的图形文件中以特定的位宽读取图像。通常情况下，读取的大多数图像均为8bit。当这些图像加载到内存中时，Matlab就将其存储在类uint8中。 ■

注意

对于索引图像来说，即使图像阵列的本身为uint8或uint16，imread函数仍然将颜色映像表读取并存储到一个双精度的浮点类型的阵列中。

(2) 图形图像文件的写入（保存）

利用**imwrite**函数可以完成图形图像文件的写入操作，其语法为： **imwrite(A, filename, fmt)** ■

imwrite(X, map, filename, fmt)

Matlab 缺省的保存方式：**uint8**数据格式。

在**Matlab**中使用的许多图像都是**8bit**，并且大多数的图像文件并不需要双精度的浮点数据。

(3) 图形图像文件信息的查询 ■

Matlab提供了**imfinfo**函数用于从图像文件中查询其信息。所获取的信息依文件类型的不同而不同，但至少包含下面的内容。 ■

文件名　　文件格式 ■

文件格式的版本号　　文件修改时间 ■

文件的字节大小　　图像的宽度（像素）

图像的长度（像素）　　每个像素的位数

图像类型（即该图像是**RGB(**真彩**)**图像、灰度图像还是索引图像）

4.3.4 图像文件的显示

Matlab 图像处理工具箱提供了一个高级的图像显示函数**imshow**。其语法格式如下，■

灰度图像 imshow(I)

imshow(I, n) ■

imshow(I, ［low high］) ■

二值图像 imshow(BW) ■

索引图像 imshow(X, map) ■

真彩色图像 imshow(RGB) ■

其中n为灰度级数目，缺省值为256。［low high］为图像数据的值域。

(1) 索引图像及其显示 ■

索引图像包括一个数据矩阵X，一个颜色映像矩阵map。其中map是一个p×3的数据矩阵，其每个元素的值均为［0, 1］之间双精度浮点型数据。map矩阵的每一行分别表示红色、绿色和蓝色的颜色值。而数据矩阵X可以是double和uint8型的，调用格式如下：

imshow(X, map)

索引图像的每一个像素都直接映射为调色板的一个入口。如果调色板包含的颜色数目多于图像颜色数目，那么额外的颜色都将被忽略；如果调色板包含的颜色数目少于图像颜色数目，则超出调色板颜色范围的图像像素都将被设置为调色板中的最后一个颜色。

如果一幅包含256色的uint8索引图像，使用一个仅有16色的调色板显示，则所有数值大于或等于15的像素都将被显示为调色板的最后一个颜色。

(2) 灰度图像及其显示

一幅灰度图像是一个数据矩阵 I，其中数据均代表了在一定范围内的颜色灰度值。Matlab 把灰度图像用数据矩阵的形式进行存储，每个元素则表示了图像中的每个像素。矩阵元素可以是double、uint8整数类型。

多数情况下，灰度图像很少和颜色映像表一起保存，但在显示灰度图像时，Matlab 仍然在后台使用系统预定义的缺省灰度颜色映像表。

imshow函数显示灰度图像 ■

- 灰度图像显示最基本的调用格式 ■

imshow(I)

Matlab 中 imshow函数使用一个灰度级系统调色板（R=G=B）来显示灰度图像。如果I是double型，若像素值为0.0，则显示为黑色，1.0则显示为白色，0.0和1.0之间的像素值将显示为灰影。

- 使用明确指定的灰度级数目

imshow(I, n)

例如：以下语句将显示一幅32个灰度级的图像I。 ■

imshow(I, 32)

- 某些情况下，可能将一些超出数据惯例范围的数据显示为一幅灰度图像

☆对于double型数组为［0,1］,对于uint8型数组为［0，255］

为了将超过数据范围的数据显示为图像，用户可以直接定义数据范围，其调用格式如下，

imshow(I,[low high])

其中**low**、**high**分别为数据的最小和最大值。

如果用户使用一个空矩阵[ ]指定数据范围， **imshow**将自动进行数据标度。

(3) RGB图像及其显示

用imshow函数显示RGB图像基本的调用格式如下，

imshow(RGB)

参数RGB是一个m×n×3的数组。对于RGB中的每一个像素（r, c），imshow显示数值（r, c, 1：3）所描述的颜色。每个屏幕像素使用24位颜色系统直接显示真彩图像，系统给每个像素的红、绿、蓝颜色分量分配8位（256级），这样就有1000多万种颜色（2^{24}）。

(4) 二进制图像及其显示 ■

显示二进制图像用如下语句，

■ **imshow(BW)**

在Matlab 中，二进制图像是一个逻辑类，仅包括0和1两个数值，像素0显示为黑色，像素1显示为白色。 ■

在显示时，也可以通过NOT(~)命令，对二进制图像进行取反，使数值0显示为白色，1显示为黑色。 ■

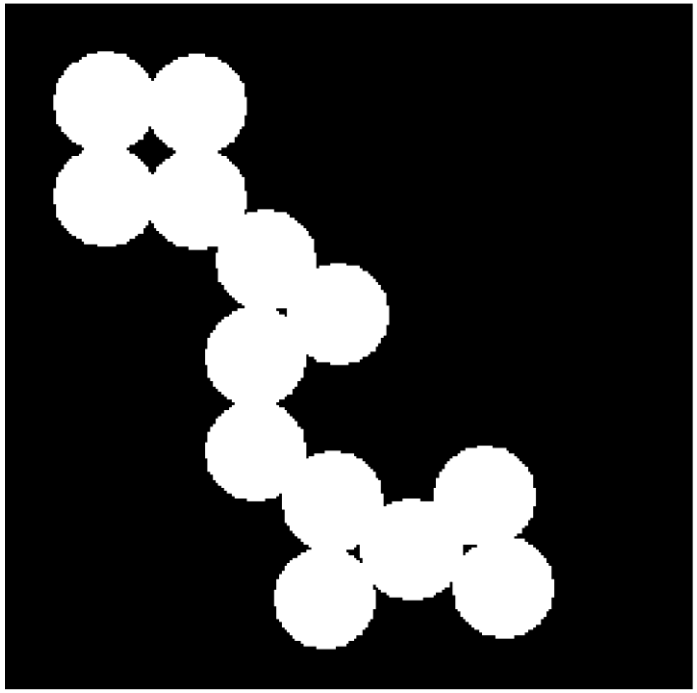
例如： **BW=imread('circles.png'); ■**

imshow(BW);

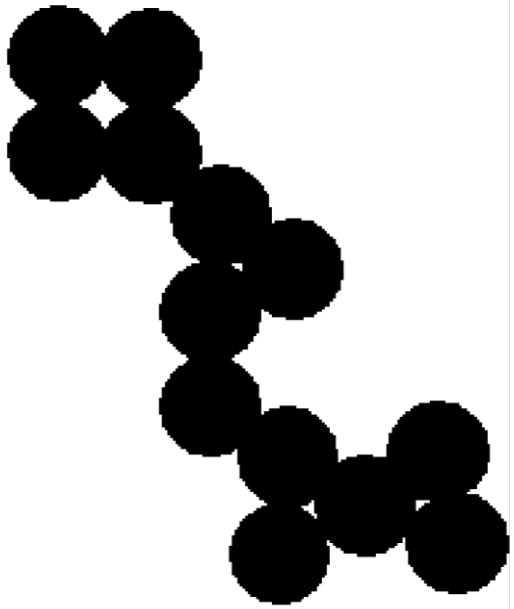
figure, imshow(~BW);

显示的结果如下图所示。 ■

二进制图像显示效果



（a）原始二进制图像



（b）取反后二进制图像

(5) 直接从磁盘显示图像 ■

通常，在显示一幅图像前首先要调用**imread**函数装载图像，将数据存储为**Matlab**工作平台中的一个或多个变量。但是，如果不希望在显示图像之前装载图像，则可以使用以下命令格式直接进行图像文件的显示， ■

imshow filename↵

其中，**filename**为要显示的图像文件的文件名。

(6) 多幅图像文件的显示

- 创建新的图像窗口，每个图像显示在一个窗口中

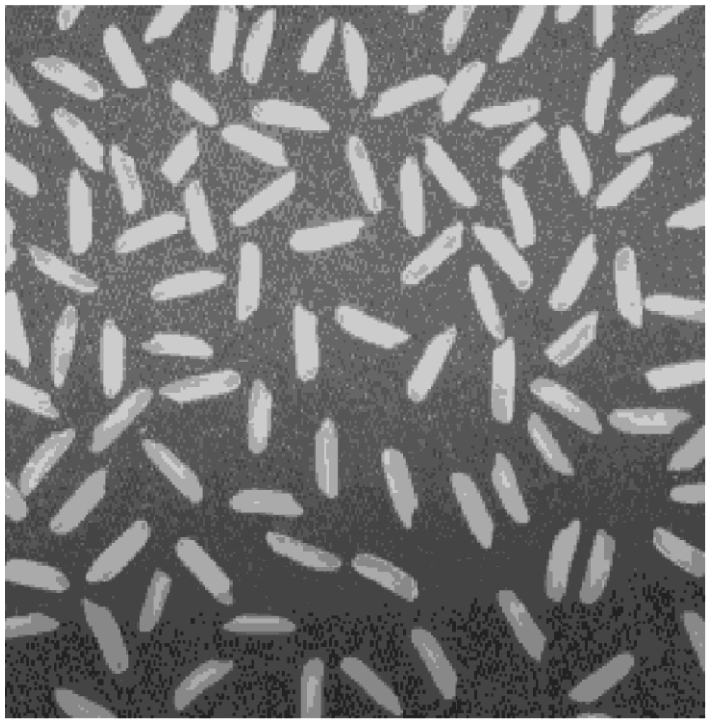
figure，imshow

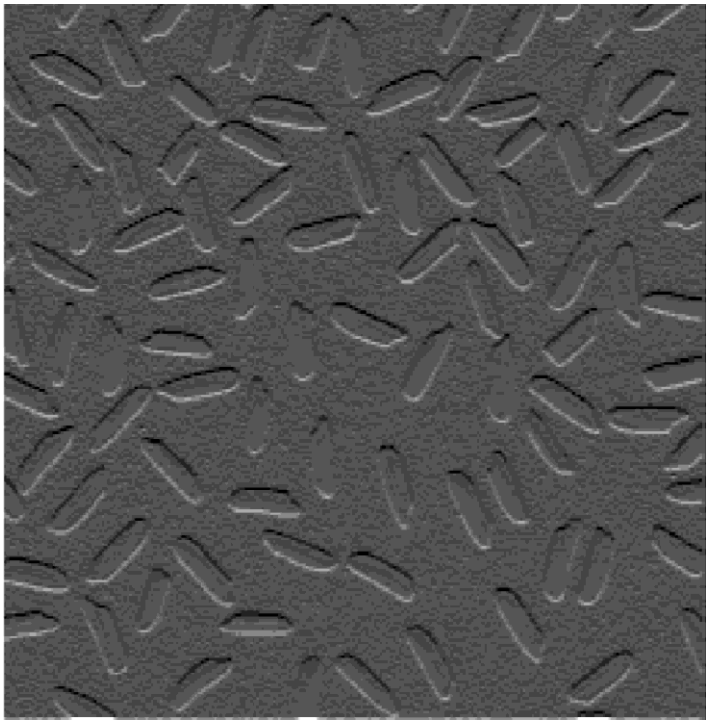
功能：新建一个图像窗口，用于显示新图像（从而不让新的图像覆盖原来图像）。

调用**imshow**函数显示图像,如下页图所示。

```
I=imread('rice.png') ■
J=filter2(［1 2;-1 -2］,I)  %用模板［1 2;-1 -2］对图像滤波 ■
imshow(I) ■
figure,imshow(J,［ ］) ■
```

用imshow显示滤波前后的图像





- 用子图、多个图像显示在一个图像窗口中

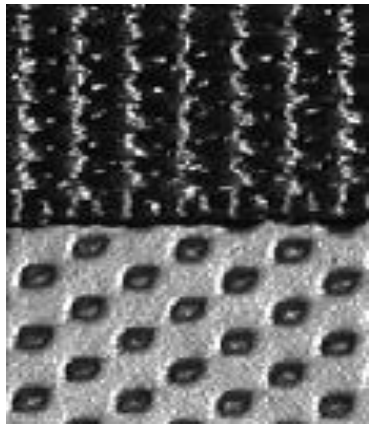
subplot(m,n,k),imshow

%绘制并显示m行n 列第k个子图

例：

```
X1=imread('rice.png');
X2=imread('coins.png');
X3=imread('bag.png');
subplot(1,3,1), imshow(X1);
subplot(1,3,2), imshow(X2);
subplot(1,3,3), imshow(X3);
```

子图显示效果



多帧图像的电影片段

若一个tif的图像文件包含有多帧的图像，我们一般用immovie来实现创建电影片段的功能，比如说，以下调用将根据多帧索引图像X创建电影片段。

```
mov=immovie(X,map)
mri=uint8(zeros(128, 128, 1, 27));
for frame=1:27
    [mri(:, :, :, frame), map]=imread('mri.tif', frame);
end
mov=immovie(mri, map);
movie(mov);
```

初始化一个包括27帧的灰度图像的文件mri.tif

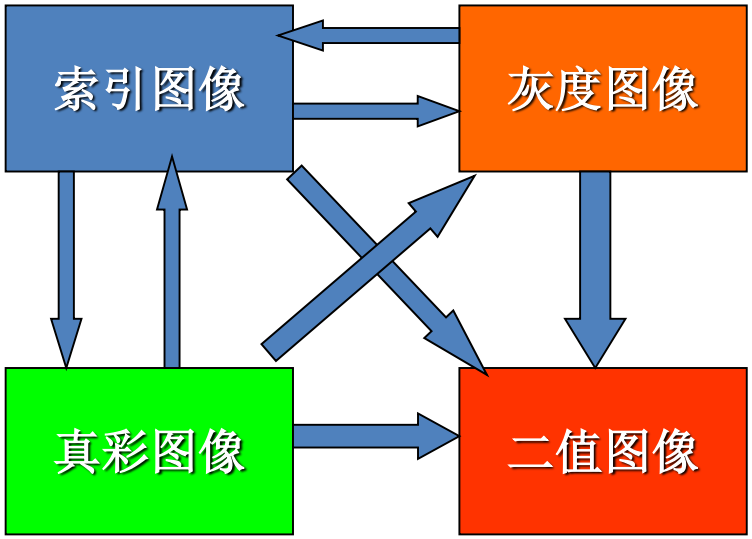
将mri.tif的每一帧读入图相应的图像帧中

4.3.5 Matlab图像类型转换

(1) 图像类型转换的必要性

例如：对于索引图像进行滤波时，必须把它转换为RGB图像，否则光对图像的下标进行滤波，得到的结果是毫无意义的。

(2) 各种类型图像的转换关系



工具箱中提供了许多图像类型转换的函数，从函数名称可以看出它们的功能。

1. dither函数 ■

功能：图像抖动，利用仅能显示少数彩色的设备显示含有丰富颜色信息图像的一种非常有用的方法。

该函数：把RGB图像转换成索引图像

把灰度图像转换成二值图像 ■

格式：**X=dither(RGB，map)**

BW=dither(I)

抖动效果图

（1） RGB图像抖动成索引图像

例：

```
I=imread('autumn.tif');
map=pink(1024);
X=dither(I,map);
imshow(I);
figure, imshow(X, map); colorbar
```

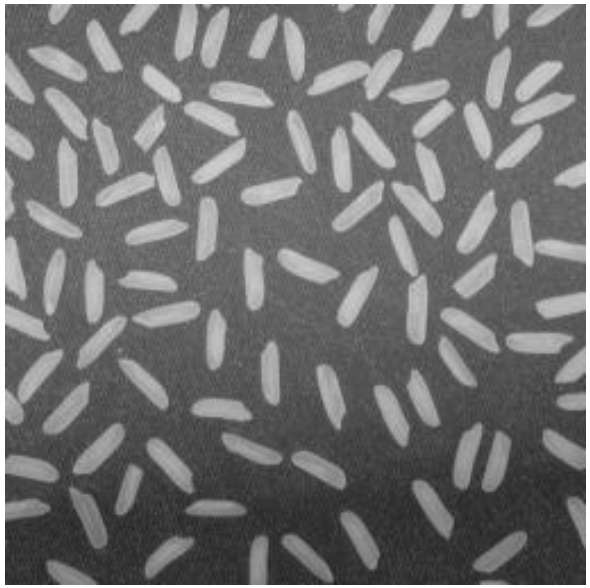
抖动效果图

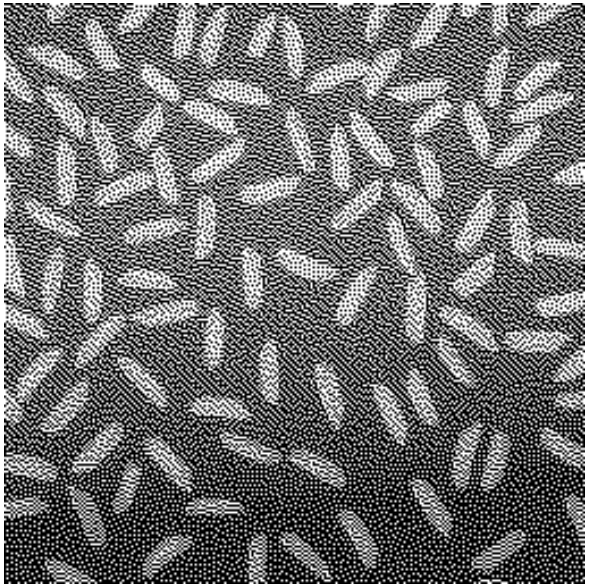
（2） 灰度图像抖动成二值图像

例： I=imread('rice.png');

```
bw=dither(I);
imshow(I);
figure, imshow(bw);
```

抖动效果图





2. gray2ind函数 ■

功能：将灰度图像转换成索引图像。 ■

格式：**[X, map]= gray2ind(I, n)**

按照指定的灰度级n 把灰度图像I转换成索引图像X, map 为gray (n) ， n的缺省值为64。

例： I=imread('cameraman.tif');

[X, map]=gray2ind(I, 16);

imshow(X, map);

figure, imshow(I);

3. grayslice函数

功能：通过设定阈值将灰度图像转换成索引色图像。

格式：**X=grayslice(I, n)**

例：

```
I=imread('cameraman.tif');
X=grayslice(I, 16);
imshow(I);
figure, imshow(X, bone(16));
```

4. im2bw函数

功能：将灰度图像、索引色图像和真彩色图像转化成二值图像。

格式：**BW=im2bw(I, level)** ■

BW=im2bw(X, map, level) ■

BW=im2bw(RGB, level) ■

level是一个归一化阈值，取值在[0,1]。

例：真彩色转换为二值图像

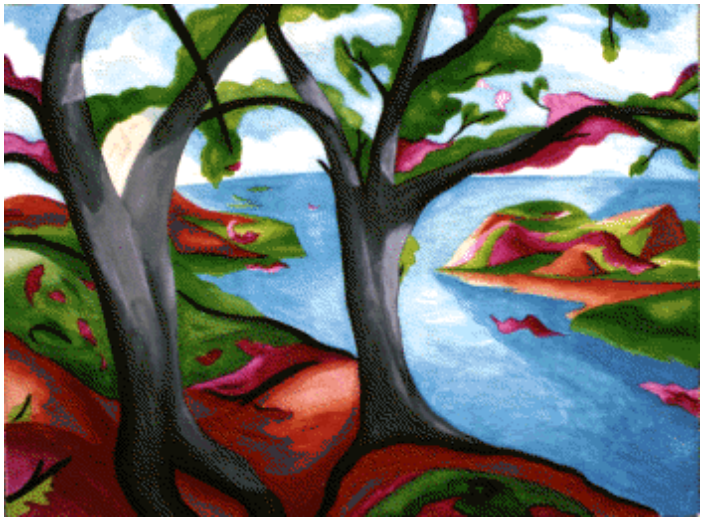
```
I=imread('autumn.tif');
X=im2bw(I, 0.5);
imshow(I);
figure, imshow(X);
```

5. ind2gray函数

功能：将索引图像转换成灰度图像。 ■

格式： **I= ind2gray(X, map)**

索引图像转换成灰度图像



6. ind2rgb函数 ■

功能：将索引色图像转换成真彩色图像。 ■

格式：**RGB=ind2rgb(X, map)**

例： ■

```
[I,map]=imread('m83.tif');
X=ind2rgb(I, map);
imshow(I, map);
figure, imshow(X);
```

7. mat2gray函数 ■

功能：将一个数据矩阵转换成一幅灰度图像。 ■

格式：I=mat2gray(A, ［amin amax］) ⌐

I=mat2gray(A) ■

8. rgb2gray函数 ■

功能：将一幅真彩色图像转换成灰度图像。 ■

格式：I= rgb2gray(RGB)⌐

例：

```
RGB=imread('autumn.tif');
X=rgb2gray(RGB);
imshow(RGB);
figure, imshow(X);
```

转换效果图

9. rgb2ind函数 ■

功能：将真彩色图像转换成索引色图像。 ■

格式： **[X, map] = rgb2ind(RGB, n)**

例：

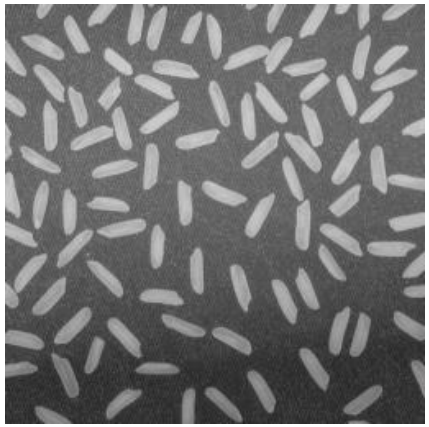
```
RGB=imread('autumn.tif');
[X, map]=rgb2ind(RGB, 128);
imshow(RGB);
figure, imshow(X, map);
```

4.3.6 图像基本运算

在利用MATLAB进行图像处理时，由于图像数据类型为unit8，而在矩阵运算中要求所有的运算变量为double型。因此必须将图像数据转换成双精度型数据。在MATLAB中，通过函数im2double( )将图像数据转换为双精度浮点型。

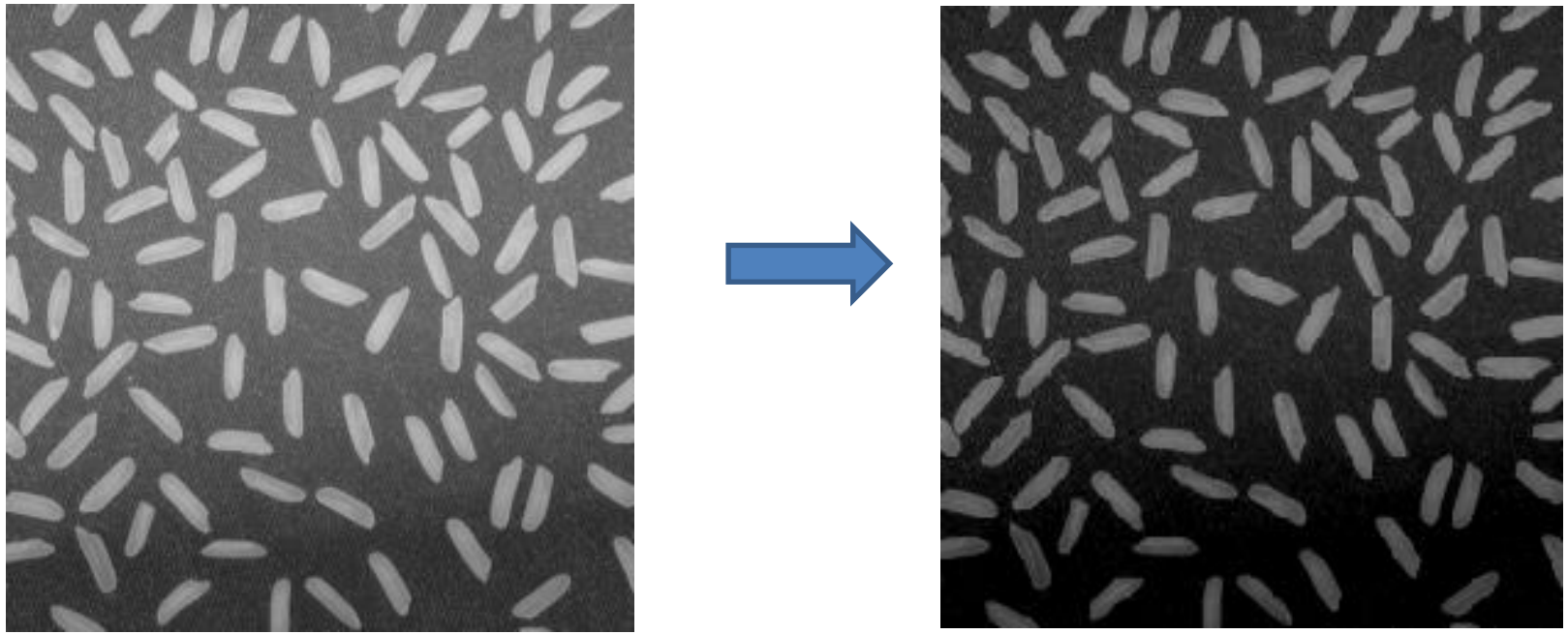
➢ 加法运算

```
I=imread('rice.png');
J=imread('cameraman.tif');
K=imadd(I,J,'uint16');
imshow(K,[]);
set(gcf,'position',[200,200,300,300]);
```



➢ 减法运算

```
I=imread('rice.png');
J=imsubtract(I,60);
imshow(J);
set(gcf,'position',[200,200,300,300]);
```



➢ 乘法运算

```
I=imread('moon.tif','tif');
J=immultiply(I,0.5);
subplot(121);
imshow(I);
subplot(122);
imshow(J);
set(gcf,'position',[200,200,600,300]);
```

➢ 除法运算

```
X=uint8([ 255 10 75; 44 225 100]);
Y=uint8([ 50 20 50; 50 50 50 ]);
Z=imdivide(X,Y)
```

```
Z =

   5   1   2
   1   5   2
```

➢ 绝对差异

```
I=imread('cameraman.tif');
J=uint8(filter2(fspecial('gaussian'), I));     %对图像滤波
K=imabsdiff(I,J);
subplot(121);
imshow(I);
subplot(122);
imshow(K,[]);
set(gcf,'position',[200,200,600,300]);
```

➢ 图像的直方图和直方图均衡化

在MATLAB中，采用函数imhist( )计算和显示图象的直方图，该函数的调用格式为：

imhist(I)：该函数绘制灰度图像的直方图。

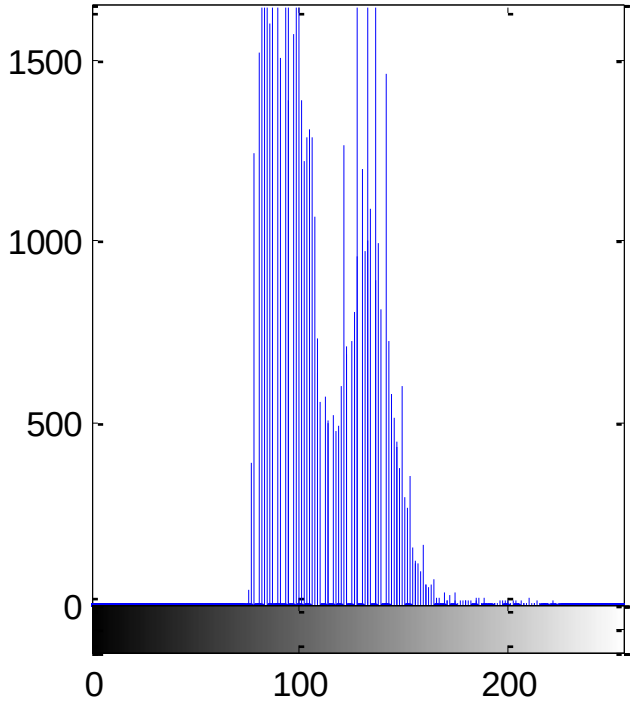
imhist(I, n)：该函数指定灰度级的数目为n。

imhist(X, map)：该函数绘制索引图像的直方图。

[counts, x]=imhist(…)：该函数返回直方图的数据，通过函数stem(x, counts)可以绘制直方图。

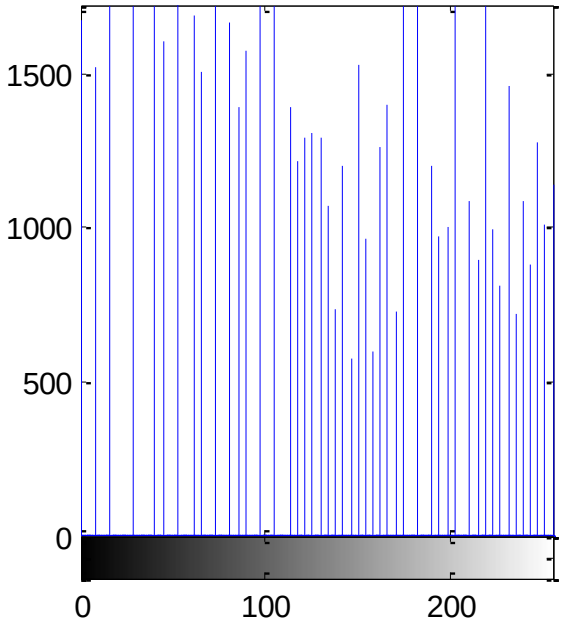
在MATLAB中，采用函数histeq( )实现直方图的均衡化。该函数可以用于灰度图像和索引图像。

```
I=imread('pout.tif');
subplot(121);
imshow(I);
subplot(122);
imhist(I);
set(gcf,'position',[200,200,600,300]);
```



采用函数histeq( )实现直方图的均衡化

```
I=imread('pout.tif');
subplot(121);
J=histeq(I);
imhist(J);
subplot(122);
imshow(J);
set(gcf,'position',[200,200,600,300]);
```



➢ 图像的对比度增强

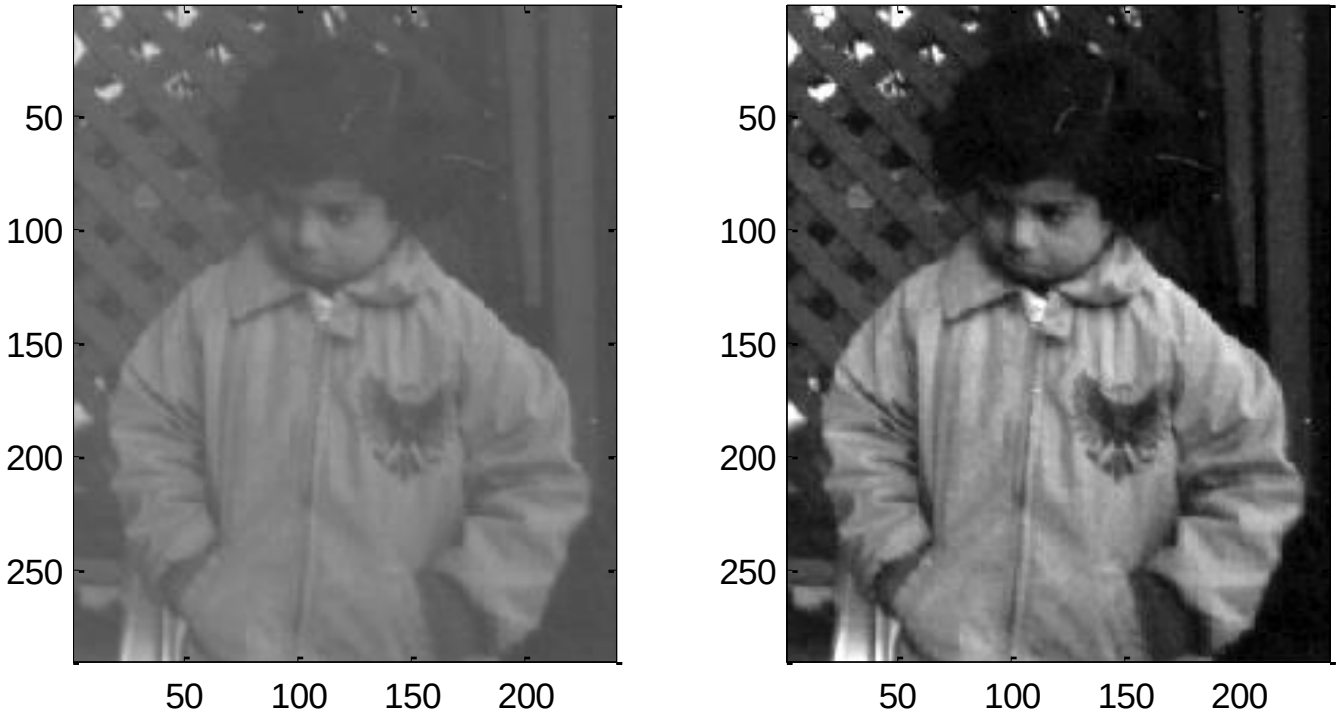
在MATLAB中，通过函数imadjust( )进行图像的对比度增强。该函数的调用格式为：

J=imadjust(I)：该函数对灰度图像I进行对比度增强。

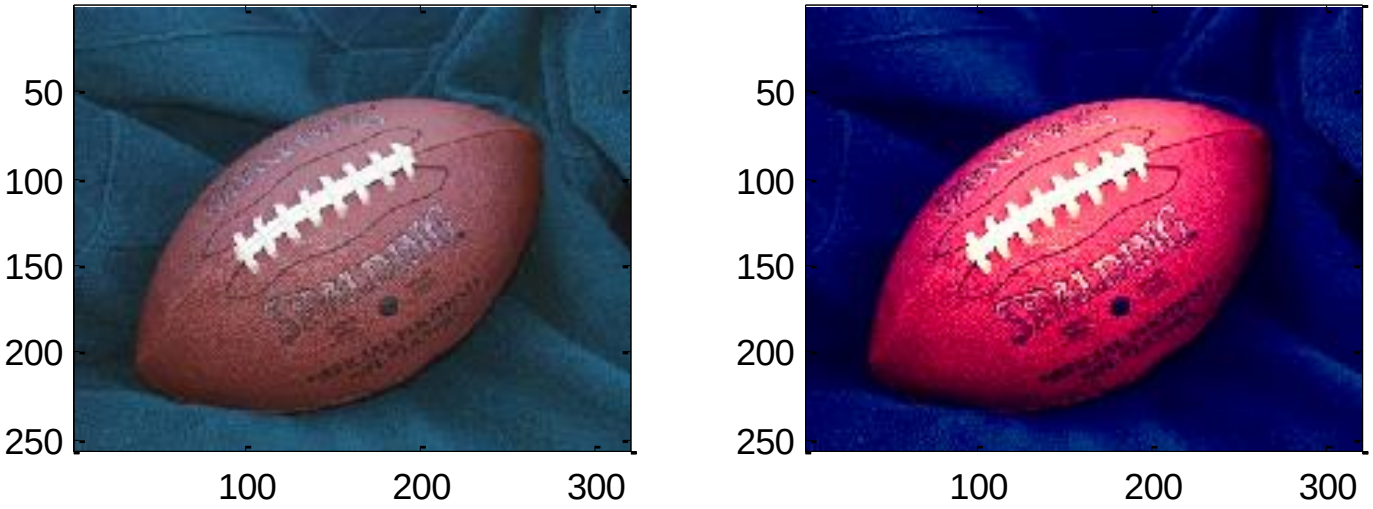
J=imadjust(I, [low_in; high_in], [low_out; high_out])：该函数中[low_in; high_in]为原图象中要变换的灰度范围，[low_out; high_out]变换后的灰度范围。

RGB2=imadjust(RGB1, …)：该函数对RGB图像进行对比度增强。

```
I=imread('pout.tif');
J=imadjust(I,[0.3 0.7],[]);
subplot(121);
subimage(I);
subplot(122);
subimage(J);
set(gcf,'position',[200,200,600,300]);
```



```
RGB1=imread('football.jpg');
RGB2=imadjust(RGB1,[0.2 .3 0; .6 .7 1],[]);
subplot(121);
subimage(RGB1);
subplot(122);
subimage(RGB2);
set(gcf,'position',[200,200,600,300]);
```



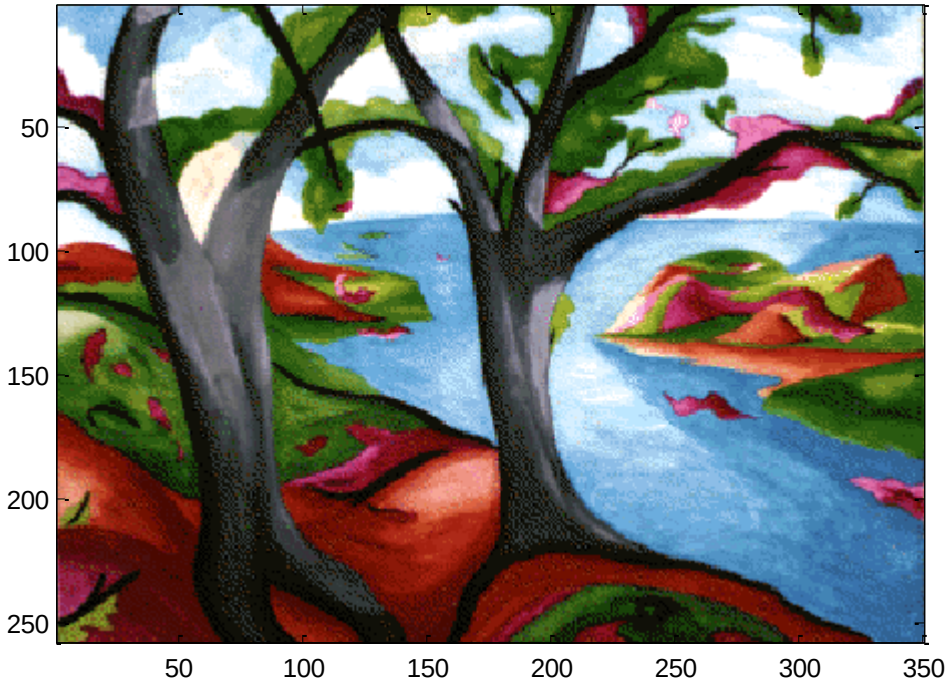
➢ 图像的插值

插值是常用的数学运算，通常是利用曲线拟合的方法，通过离散的采样点建立一个连续函数来逼近真实的曲线，用这个重建的函数便可以求出任意位置的函数值。在MATLAB中通过插值，可以实现图像的缩放和旋转。

函数**imresize( )**采用插值的方法来改变图像的大小。

函数**imrotate( )**进行图像的旋转。

```
[X,map]=imread('trees.tif','tif');
[Y,newmap]=imresize(X,map,0.5);
figure;
subimage(X,map);
figure;
subimage(Y,newmap);
```



```
A=imread('football.jpg','jpg');
B=imrotate(A,-20,'nearest');
figure;
subplot(121);
imshow(A);
subplot(122);
imshow(B)
set(gcf,'position',[200,200,600,300]);
```

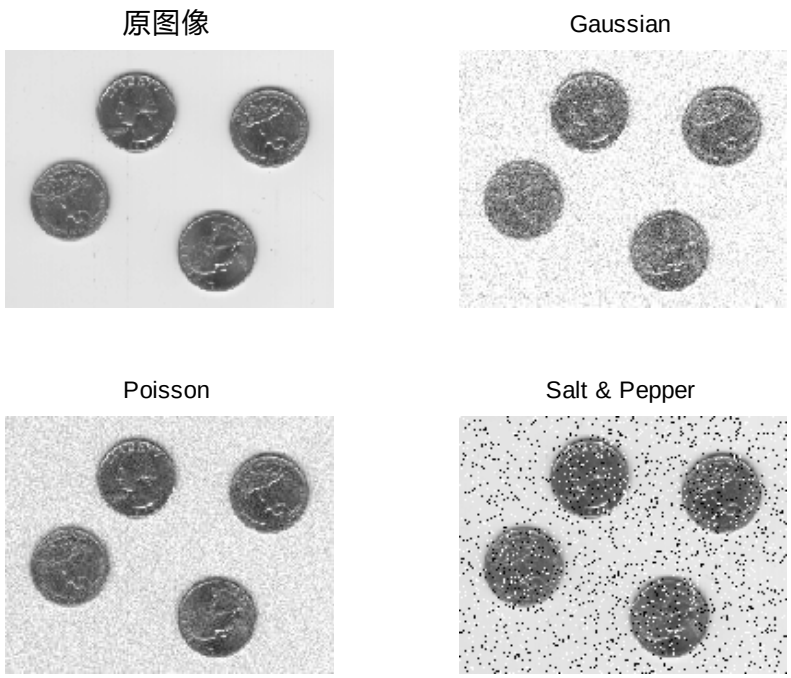
➢ 图像中添加噪声

在MATLAB中，可以通过函数imnoise( )给图像添加噪声，该函数的调用格式为：

J=imnoise(I, type)：该函数对图像I添加类型为type的噪声。参数Type对应的噪声类型如下：'gaussian'为高斯白噪声；'localvar'为0均值白噪声；'poisson'为泊松噪声；'salt & pepper'为椒盐噪声；'speckle'为乘性噪声

J=imnoise(I, type, parameters)：该函数对类型为type的噪声的属性进行设置。

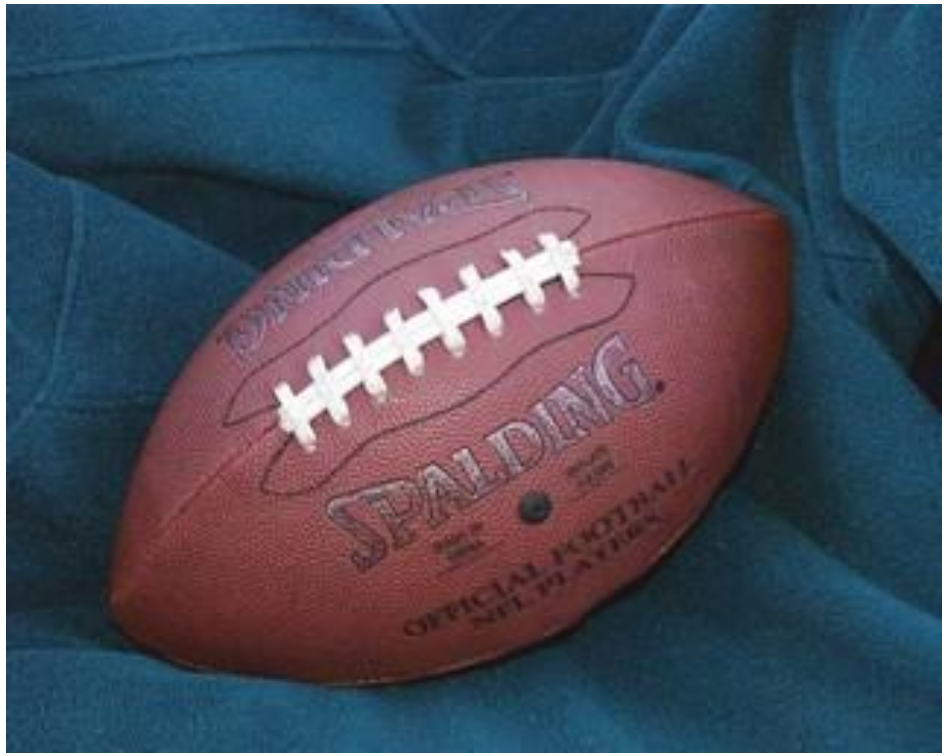
```
I=imread('eight.tif','tif');
J1=imnoise(I,'gaussian',0.1);
J2=imnoise(I,'poisson');
J3=imnoise(I,'salt & pepper',0.1);
figure;
subplot(221),imshow(I),title('原图像')
subplot(222),imshow(J1),title('Gaussian')
subplot(223),imshow(J2),title('Poisson')
subplot(224),imshow(J3),title('Salt & Pepper');
```



➢ 图像剪切

通过函数imcrop( )实现图像的剪切，获取图像的一部分(矩形区域)

```
I=imread('football.jpg','jpg');
J=imcrop(I,[40,40,120,150]);
figure;
imshow(I);
figure;
imshow(J);
```



➢ 图像的变换技术

在数字图像处理中，图像的变换包括：二维快速傅立叶变换和反变换、二维离散余弦变换和反变换。下面分别进行介绍。

➢ 图像的傅里叶变换

在MATALB中，采用函数fft2( )计算图像的二维快速傅立叶变换，该函数的调用格式为：

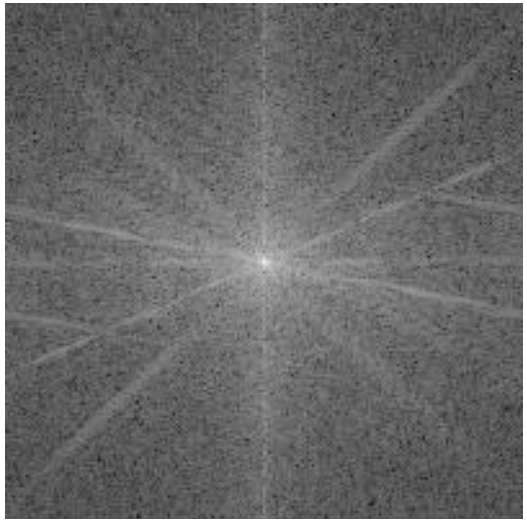
Y=fft2(X)：该函数计算图像数据X的二维傅立叶变换。

Y=fft2(X, m, n)：该函数通过补0，来指定数据的大小。

ifft2( )可以实现傅里叶反变换

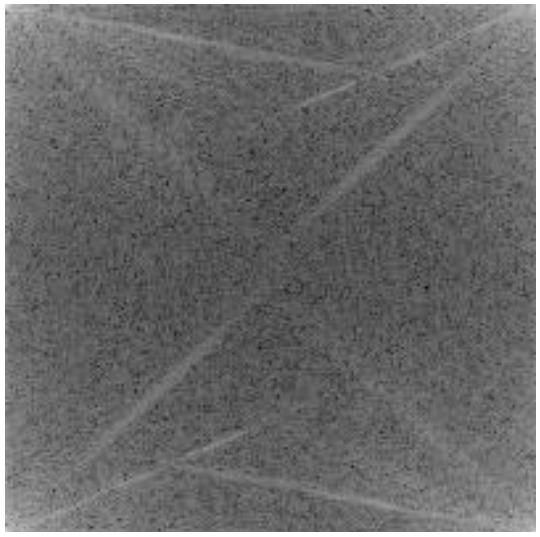
```
I=imread('cameraman.tif','tif');
J=fft2(I,256,256);
K1=fftshift(J);
K2=log(abs(K1));
figure;
subplot(121);
imshow(I);
subplot(122);
imshow(K2,[]);
set(gcf,'position',[200,200,600,300]);
```

傅里叶变换



```
I=imread('cameraman.tif','tif');
J=fft2(I,256,256);
K=ifft2(J,256,256);
figure;
subplot(121);
imshow(log(abs(J)),[]);
subplot(122);
imshow(K,[]);
set(gcf,'position',[200,200,600,300]);
```

傅里叶变换
傅里叶反变换



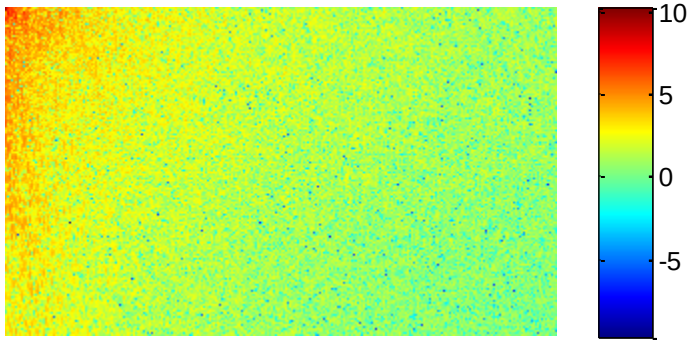
➢ 图像的离散余弦变换

离散余弦变换（Discrete Cosine Transform）是采用不同频率和幅值的余弦函数来逼近图像。

离散余弦变换常用来进行图像的压缩，例如JPEG格式的图像就采用了离散余弦变换进行压缩。

在MATLAB中，采用函数dct2( )进行二维离散余弦变换，采用函数idct2( )进行二维离散余弦变换的反变换 。

```
RGB=imread('autumn.tif');
I=rgb2gray(RGB);
J=dct2(I);
figure;
imshow(log(abs(J)),[]),colormap(jet),colorbar;
J(abs(J)<10)=0;
K=idct2(J);
figure;
imshow(K,[0 255]);
% set(gcf,'position',[200,200,700,300]);
```



➢ 图像增强技术

数字图像的增强是图像处理中的一个重要研究内容之一，是图像处理的一项基本技术。下面介绍如何获取图像的像素值及其统计，再介绍通过图像的数字滤波进行图像的增强。

像素值及其统计

在MATLAB中，采用函数impixel( )来获取图像的像素值。该函数的常用调用格式为：

p=impixel(I)：该函数通过鼠标单击获取灰度图像中一点的像素值。

p=impixel(X, map)：该函数通过鼠标单击获取索引图像中一点的像素值。

p=impixel(RGB)：该函数通过鼠标单击获取RGB图像中一点的像素值。

p=impixel(I, c, r)：该函数获取灰度图像中，行为c，列为r的像素点的像素值。

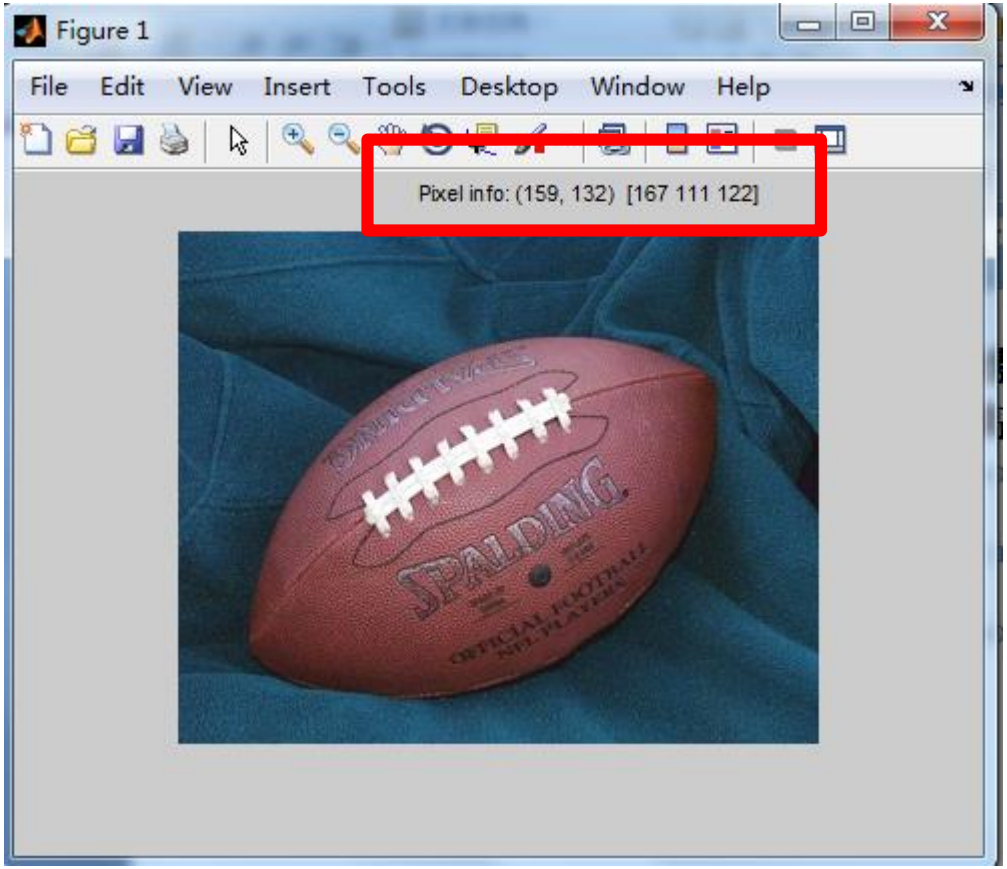
p=impixel(X, map, c, r)：该函数获取索引图像中，行为c，列为r的像素点的像素值。

p=impixel(RGB, c, r)：该函数获取RGB图像中，行为c，列为r的像素点的像素值。

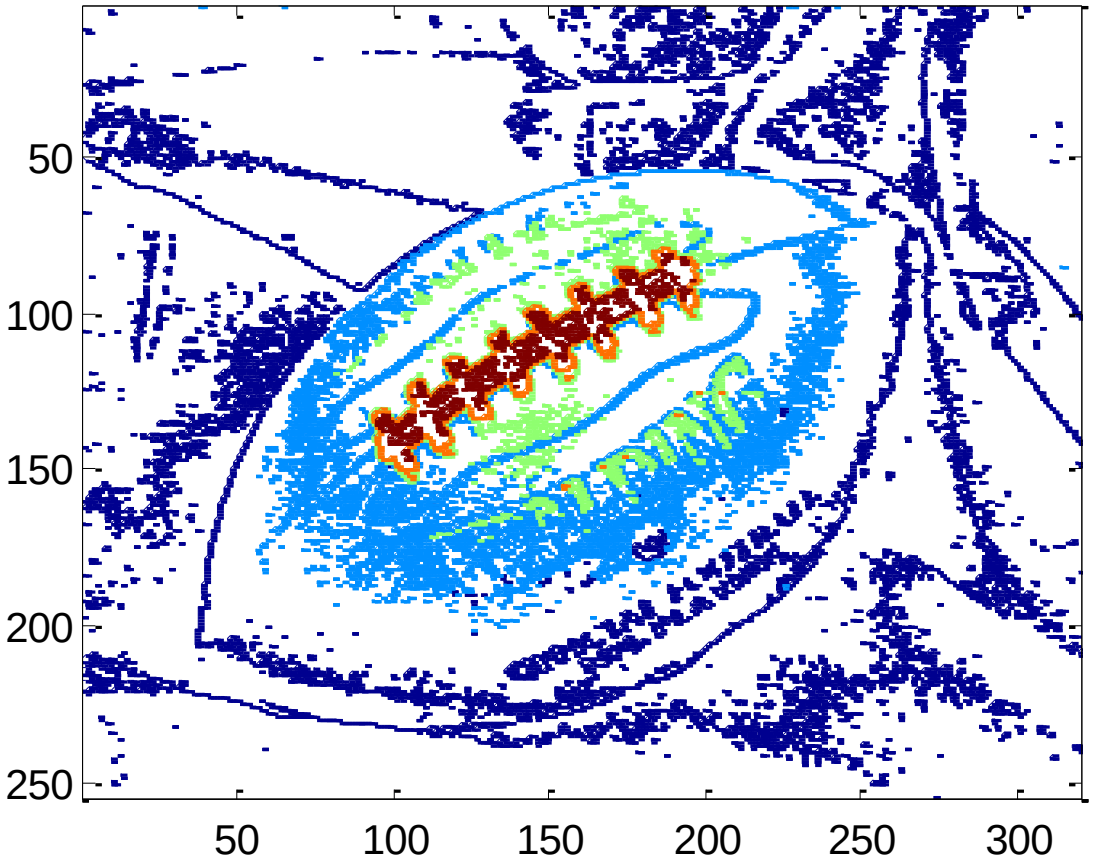
利用函数impixelinfo( )获取图像中任意点的像素值。

利用函数imcontour( )可以绘制灰度图像的等高线 。

```
imshow('football.jpg');
h=impixelinfo;
set(h,'position',[200 320 330 20]);
```



```
RGB=imread('football.jpg');
I=rgb2gray(RGB);
figure;
imcontour(I);
set(gcf,'position',[200,200,400,350]);
```

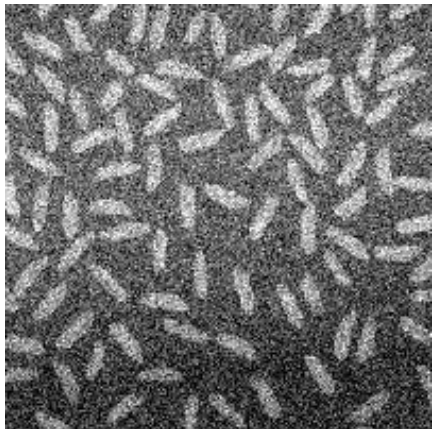


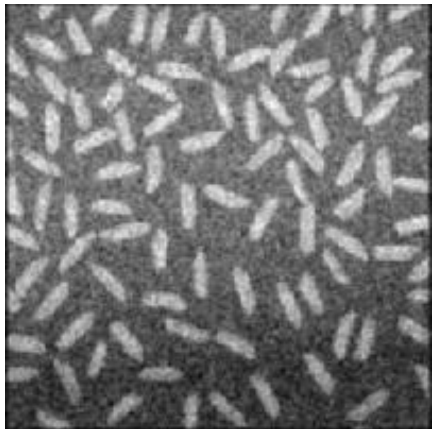
➢ 图像的滤波

对于含有噪声的图像，可以对图像进行滤波，使图像变的更清晰。常用的滤波方法有邻域平均法、中值滤波法和自适应滤波法。

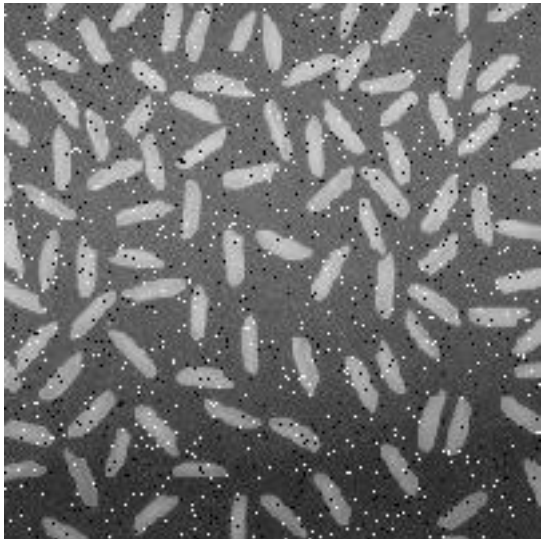
在MATLAB中，采用函数**medfilt2( )**对图像进行中值滤波。采用函数**wiener2( )**进行自适应滤波，根据图像的局部均值和方差进行自动调整，而且该函数还可以估计噪声的类型。

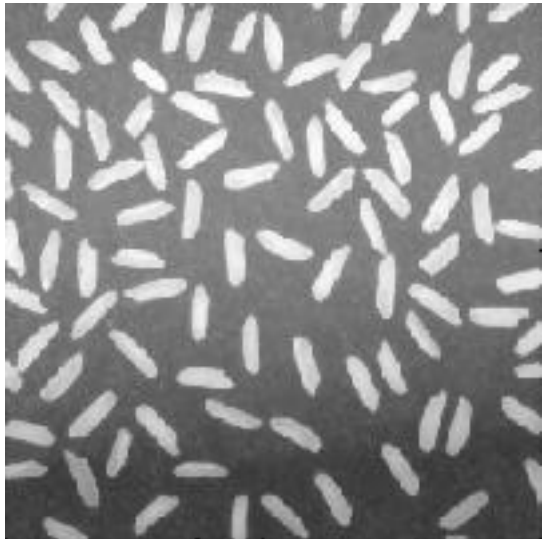
```
I=imread('rice.png');
I=imnoise(I,'Gaussian',0,0.02);
h=[1 1 1;1 1 1;1 1 1];
h=h/9;
J=conv2(I,h);
figure;
subplot(121);
imshow(I);
subplot(122);
imshow(uint8(J),[])
set(gcf,'position',[200,200,600,300]);
```



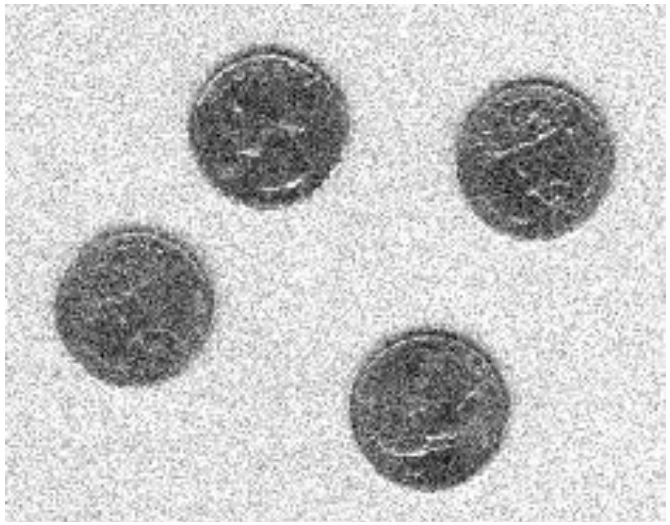


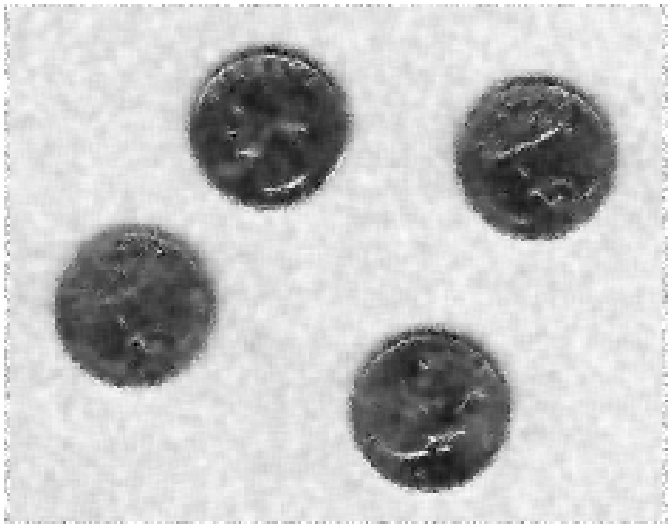
```
I=imread('rice.png');
I=imnoise(I,'salt & pepper',0.03);
J=medfilt2(I,[3 3]);
figure;
subplot(121);
imshow(I);
subplot(122);
imshow(uint8(J),[])
set(gcf,'position',[200,200,600,300]);
```





```
I=imread('eight.tif');
I=imnoise(I,'Gaussian',0,0.01);
J=wiener2(I,[5 5]);
figure;
subplot(121);
imshow(I);
subplot(122);
imshow(uint8(J),[]);
set(gcf,'position',[200,200,700,300]);
```





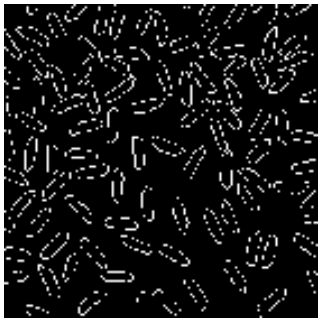
➢ 图像的边缘检测

在进行图像的分析和处理时，图像的边缘包含许多中重要的信息，可以利用边缘检测来对图像进行分割。

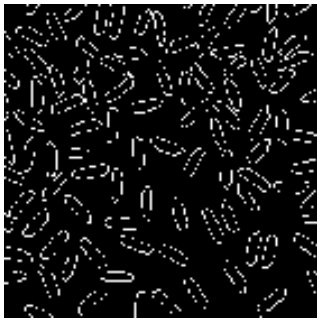
在MATLAB中，采用函数edge( )来对图像的边缘进行检测。在进行边缘检测时，常用的算子有Sobel算子、Prewitt算子、Roberts算子、LOG算子和Canny算子等。

```
I=imread('rice.png');
J1=edge(I,'Sobel');
J2=edge(I,'prewitt');
J3=edge(I,'Roberts');
figure;
subplot(221),imshow(I);
subplot(222),imshow(J1);title('Sobel');
subplot(223),imshow(J2);title('Prewitt');
subplot(224),imshow(J3);title('Roberts');
```

Sobel



Prewitt



Roberts