第3章 chapter 3

图像空间域处理与邻域操作

本章学习图像的空间域处理与邻域操作等内容。空间域的概念是与频域的概念相对应的，频域处理首先对图像进行傅里叶变换或者离散余弦等变换，然后进行处理。图像空间域处理时把图像看作关于像素位置的函数，一般使用简单的代数几何统计等方法，更多的时候使用基于导数梯度思想的邻域操作。

图像的邻域操作是图像处理中的基本操作之一，本章各节的内容都与邻域操作有关，都是基于邻域内像素点的颜色相关性进行的。

3.1 图像增强

图像增强是对图像进行操作，是在原有图像的基础上进行的，目的是得到视觉效果更好或者更有用的新图像。狭义上的图像增强就是加强灰度图像的明暗对比度，广义上的图像增强还包括图像模糊处理以及彩色图像增强等。本节介绍广义图像增强。

事实上，在前面章节已经介绍过图像增强，例如通过图像加减乘除运算对图像进行增强或者弱化图像。下面集中介绍一些基于颜色对比的灰度图像增强方法，以及 RGB 图像与 HSV 图像增强方法。

3.1.1 灰度调整

有时增加灰度图像的明暗对比度，灰度图像就变得更加清楚。增加明暗对比度的一种常用方法是灰度调整方法。灰度调整方法是基于灰度直方图的一种图像增强方法。灰度直方图就是类似下面例 3-1 绘制出的灰度图像颜色统计图。有关灰度直方图的更具体的内容将在第 5 章中详细介绍。

【例 3-1】 统计  (小狮子)灰度图像数据，计算从 0～255 种颜色每种颜色的像素点个数。

编写下面程序：

```
A=imread('D:\shizi1.bmp');  %图像 shizi1.bmp 存放在作者机器的 D 盘根目录下
A1=rgb2gray(A);             %变为灰度图像,大小为 16×16
A2=floor(double(A1));       %变为双精度浮点数后再取整
```

```
N=zeros(256);                    %生成一个 256 行 256 列的元素全是 0 的矩阵(二维数组)
for i=1:16                       %嵌套循环
    for j=1:16
        k=A2(i,j);               %取出灰度图像的(i,j)位置的颜色值
        N(k+1)=N(k+1)+1;         %把颜色值为 k 的像素点数加 1,存在数组 N 的第 k+1 个元素中
    end
end
bar(N)                           %绘制统计用柱状图
```

程序中,语句 N=zeros(256)创建了一个 256 个一维向量数组,每个元素的值为 0。使用两个嵌套循环语句访问了图中的 256 个像素,根据像素的颜色值 k(0≤k≤255),那么把数组 N 中记载的具有 k+1 颜色的点数增加 1。之所以用 k+1,是因为 MATLAB 数组的下标从 1 开始,而不是从 0 开始,在这里就使用 N(1)表示颜色为 0 的像素的个数,N(2)表示颜色为 1 的像素的个数……N(256)表示颜色为 255 的像素的个数。在 N 中存储 256 种颜色每种颜色的像素点个数。

最后使用 bar(N)绘制出了图 3-1 所示图形。

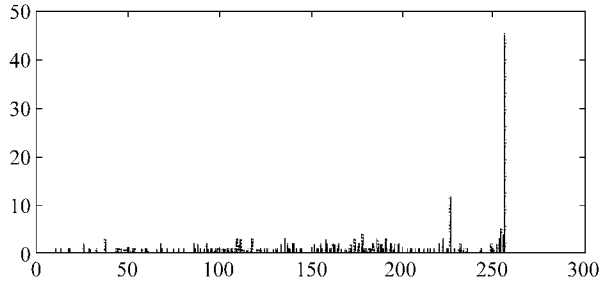


图 3-1 小狮子图像 0～255 种颜色每种颜色的像素点个数

从图 3-1 可以看出,图像中颜色是 255 的像素最多,大约 40 多个。80～200 之间的像素点个数比较起来多一些,不过,都没有超过 10 个。由于该图像小,所以得到的统计特性不是很明显。下面例 3-2 对一个较大的图像进行了统计。

【例 3-2】 统计图像 pout.tif 的每个灰度颜色的像素点个数。

图像 pout.tif 是 MATLAB 自带的实验用图像(图 3-3(a)),修改例 3-1 中的程序如下:

```
A=imread('pout.tif');
A1=floor(double(A));
S=size(A1);
N=zeros(256);
for i=1:S(1)
    for j=1:S(2)
        k=A1(i,j);
        N(k+1)=N(k+1)+1;
    end
end
bar(N)
```

运行程序得到图 3-2 所示的统计图。

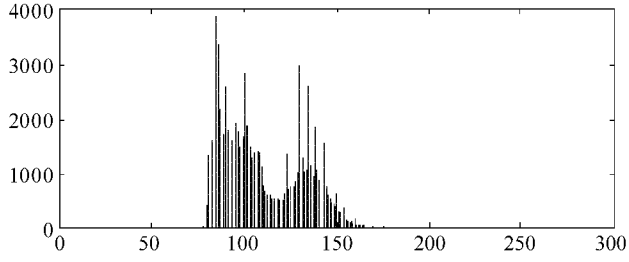


图 3-2　图像 pout. tif 0～255 种颜色每种颜色的像素点个数统计图

从图 3-2 可以看出图像的颜色主要集中在 90～120 之间。现在讨论如何把位于 90～120 之间的颜色值离散开，按比例离散为分布在 0～255 之间的颜色值。同时把颜色值小于 90 的像素颜色置 0，把颜色值大于 120 的像素颜色置为 255。

设位于 90～120 之间的颜色值为 x，要变成 0～255 之间的颜色值为 y，让 y 与 x 满足：

$$\frac{x-90}{120-x}=\frac{y-0}{255-y} \tag{3-1}$$

根据式(2-1)能够得到 y 与 x 的关系：

$$y=\frac{255(x-90)}{120-90} \tag{3-2}$$

使用式(3-2)能够把 90～120 之间的颜色值离散到 0～255 之间。不过，在对原有矩阵使用该变换后，原先矩阵中小于 90 的元素会变为负数，大于 120 的元素会超过 255。所以在变换后，可以把变为负数的元素置为 0，超过 255 的元素置为 255。

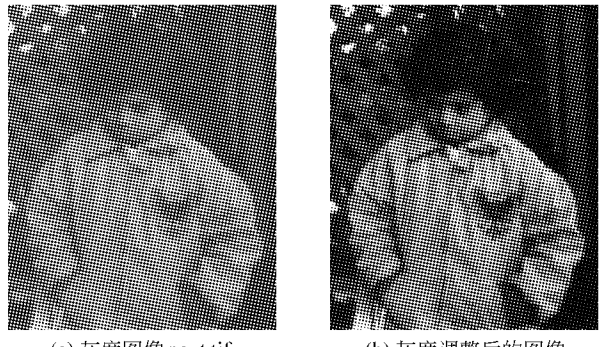
【例 3-3】 根据图像 pout. tif 的颜色分布情况调整图像灰度值，增强该图像的明暗对比度。

根据式(3-2)以及上面的分析，设计程序如下：

```
A=imread('pout.tif');
A2=double(A);
A3=(A2-90)*255/30;
s=size(A3);
for i=1:s(1)
    for j=1:s(2)
        if A3(i,j)<0
           A3(i,j)=0;
        end
        if A3(i,j)>255
           A3(i,j)=255;
        end
    end
end
subplot(1,2,1)
```

```
imshow(A)
subplot(1,2,2)
image(A3)
```

程序运行结果如图 3-3 所示。



(a) 灰度图像 pout.tif　　(b) 灰度调整后的图像

图 3-3　图像 pout. tif 灰度调整前后比较

上面介绍的是一种基本的灰度调整方法。可以基于这种思想设计新的灰度调整方法，例如研究人员已经提出的直方均等化方法、去相关拉伸方法等。这些方法可以参考其他文献。

3.1.2　灰度调整函数

MATLAB 提供了函数 imadjust、histeq、adapthiteq、brighten 等进行灰度调整。可以在以后的研究工作中直接使用这些函数调整图像灰度(颜色)对比度。

1. imadjust 函数

函数 imadjust 可以将图像的灰度值调整到一个指定的范围，实现类似例 3-3 的功能，以增强图像的明暗对比。

【例 3-4】 使用函数 imadjust 对图像进行灰度调整。

编写如下程序：

```
A1=imread('pout.tif');
B1=imadjust(A1,[0.2 0.5],[0,1]);
A2=imread('cameraman.tif');
B2=imadjust(A2,[0,0.2],[0.5,1]);
subplot(1,4,1)
imshow(A1)
subplot(1,4,2)
imshow(B1)
subplot(1,4,3)
imshow(A2)
```

```
subplot(1,4,4)
imshow(B2)
```

程序运行结果如图 3-4 所示。

(a) pout.tif　(b) 灰度调整后　(c) cameraman.tif　(d) 灰度调整后

图 3-4　使用 imadjust 函数对图像进行灰度调整

程序中语句 B1＝imadjust(A1,[0.2 0.5],[0,1])的第一个参数是要处理的矩阵，第二个参数用来限制输入范围，如果原来图像的颜色值是 0～255，那么把小于 255×0.2 的颜色值置为 0，把大于 255×0.5 的值置为 255，再把其他介于中间的值映射到第三个参数决定的区间。这个语句的第三个参数为[0 1]，那么该例题就映射到 0～255。

调用函数 imadjust 时，如果要映射到 0～255，那么第三个参数可以省略，省略时默认为[0 1]。例如，把程序中的语句

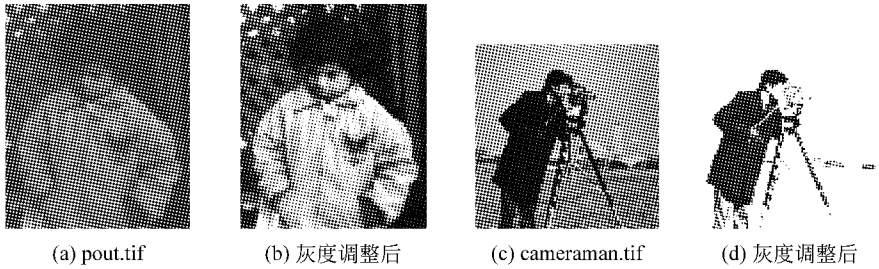
```
B1=imadjust(A1,[0.2 0.5],[0,1])
```

修改为：

```
B1=imadjust(A1,[0.2 0.5])
```

运行结果与修改前一样。

如果使用语句 B1＝imadjust(A1,[0.3,0.6])与 B2＝imadjust(A2,[0,0.2])没有输出区间，那么默认都映射到[0 1]。绘制出的图形如图 3-5 所示。



(a) pout.tif　(b) 灰度调整后　(c) cameraman.tif　(d) 灰度调整后

图 3-5　imadjust 函数默认输出范围为[0,1]

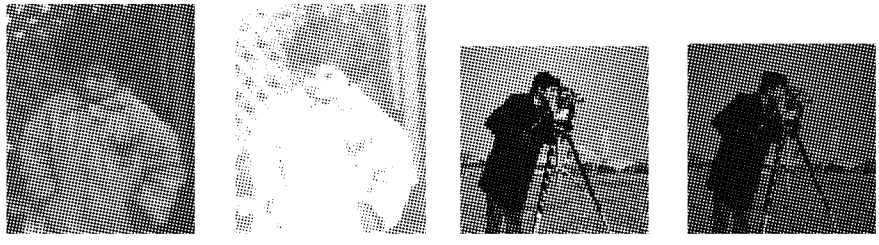
调用函数 imadjust 时，如果要把原图像的所有颜色值都映射到新图像，那么第二个参数可以写为[]。例如，把例 3-4 程序中的语句

```
B1=imadjust(A1,[0.2 0.5],[0,1]);与 B2=imadjust(A2,[0,0.2],[0.5,1]);
```

修改为：

```
B1=imadjust(A1,[0.1,0.4]);B2=imadjust(A2,[],[0,0.5]);
```

那么绘制出图 3-6 所示图形，其中，A2 是把所有的颜色值都映射到 0～128，所以图像变黑了。



(a) pout.tif　(b) 灰度调整后　(c) cameraman.tif　(d) 灰度调整后

图 3-6　imadjust 函数默认输入范围为[0,1]

【思考题】 图 3-6(b)为什么变亮了？

函数 imadjust 的调用形式还有：

```
J=imadjust(I,[LOW_IN HIGH_IN],[LOW_OUT HIGH_OUT],GAMMA)
```

第四个参数 GAMMA 是用来决定映射方式的，省略时为 1，是线性映射，与例 3-3 原理相同。当参数 GAMMA 不等于 1 时不是线性映射。

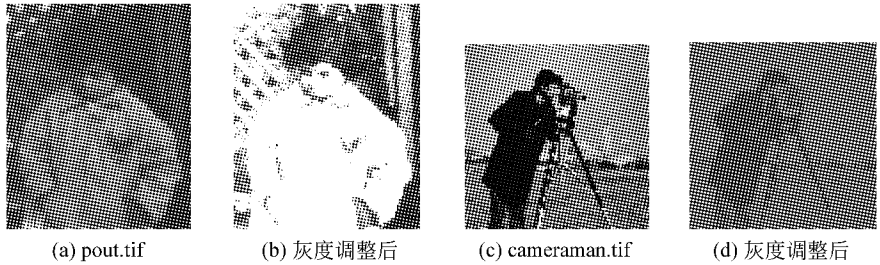
例如，如果把绘制图 3-6 的程序语句

```
B1=imadjust(A1,[0.1,0.4]); B2=imadjust(A2,[],[0,0.5]);
```

修改为：

```
B1=imadjust(A1,[0.1,0.4],[0 1],3); B2=imadjust(A2,[],[0,0.5],0.1);
```

那么绘制出图 3-7 所示图形。可以看出参数 GAMMA 大于 1 时，加强了暗色的值的输出；当参数 GAMMA 小于 1 时，加强了亮色的值的输出。



(a) pout.tif　(b) 灰度调整后　(c) cameraman.tif　(d) 灰度调整后

图 3-7　imadjust 函数的第四个参数决定图像的亮度

2. stretchlim 函数

函数 stretchlim(A)是用来计算灰度矩阵 A 的最佳输入区间，即 imadjust (I,[LOW_

IN HIGH_IN],[LOW_OUT HIGH_OUT])中的第二个参数[LOW_IN HIGH_IN]。所谓最佳是指按照这个区间输入的话，图像的灰度对比度最大。

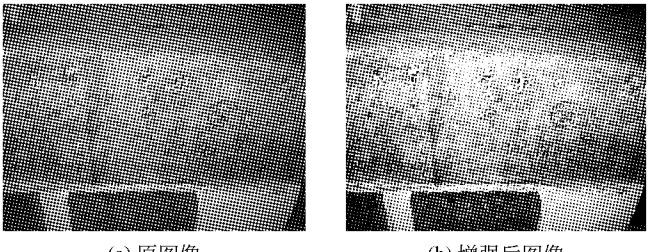
例如，使用该函数计算例 3-3 中使用的图像的最佳输入区间为[0.3059 0.6314]；cameraman.tif 的最佳输入区间为[0.0353 0.8039]。而下面例 3-5 中的图像“D:\imageprocess\IMG_7420.jpg”的最佳截取输入区间为[0.1608 0.6941]。

如果要增强图像灰度对比度，可以使用函数 stretchlim 计算出最佳输入区间，然后再调用 imadjust 函数进行图像增强。

【例 3-5】 使用函数 stretchlim 与 imadjust 对图像进行增强操作。

```
I=imread('D:\imageprocess\IMG_7420.jpg');
K=rgb2gray(I);
G=stretchlim(K);
J=imadjust(K,G);
subplot(1,2,1); imshow(I)
subplot(1,2,2); imshow(J)
imwrite(J,'4.jpg','Quality',100)
```

该图像的运行效果如图 3-8 所示，除了显示外，语句 imwrite(J,'4.jpg','Quality',100)把处理后的图像以名字 4.jpg 存储在当前工作目录下。“'Quality',100”是图像质量参数。



(a) 原图像　　(b) 增强后图像

图 3-8　使用函数 stretchlim 与 imadjust 对图像进行增强操作

其实，MATLAB 已经提供了一个函数 histeq 来自动完成图像灰度对比增强运算。

3. histeq 函数

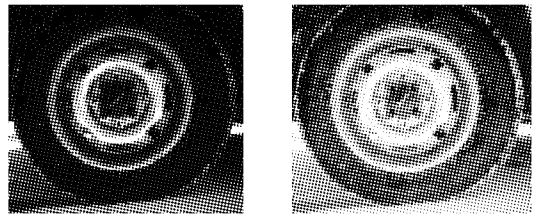
函数 histeq 能够自动完成图像灰度调整，一般用来增强图像的灰度对比度。

【例 3-6】 使用函数 histeq 增强图像的灰度对比度。

使用下面程序能够得到图 3-9 所示效果。

```
I=imread('tire.tif');
J=histeq(I);
subplot(1,2,1);imshow(I)
subplot(1,2,2);imshow(J)
```

如果把程序中的读入图像改为 D:\imageprocess\IMG_7420.jpg(即图 3-8(a))，再加

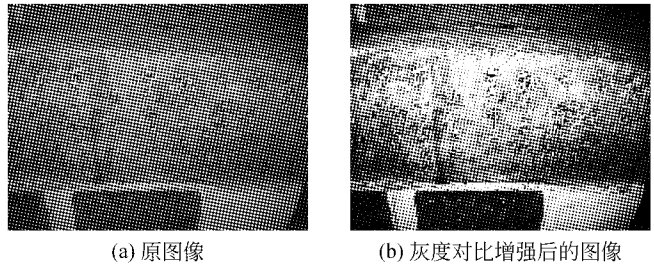


(a) 原图像　　(b) 灰度对比增强后的图像

图 3-9 使用 histeq 函数增加图像灰度对比度(1)

上一个变为灰度图像的语句,如下所示,那么处理后的效果如图 3-10(b)所示。

```
I=imread('D:\imageprocess\IMG_7420.jpg');
J=rgb2gray(I);
J1=histeq(J);
subplot(1,2,1);imshow(I)
subplot(1,2,2);imshow(J1)
```



(a) 原图像　　(b) 灰度对比增强后的图像

图 3-10 使用 histeq 函数增加图像灰度对比度(2)

4. brighten 函数

【例 3-7】 使用函数 brighten 增加灰度图像的亮度。

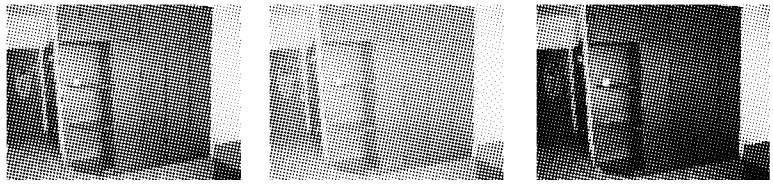
设计下面程序:

```
J=imread('D:\imageprocess\IMG_5943.jpg');
A=rgb2gray(J);
imshow(A)
figure,imshow(A)
brighten(0.6)
figure,imshow(A)
brighten(-0.6)
```

程序运行后结果如图 3-11 所示。语句 brighten(0.6)的参数大于 0 小于 1,所以显示出的图像 I 变亮;语句 brighten(−0.6)的参数小于 0 大于−1,所以显示出的图像 A 变暗。

【注】 *函数 brighten 放在函数 imshow 的后面。*

【思考题】 例 3-7 中的函数 brighten 使图像变亮或者变暗的规则是什么?是否每个位置上的颜色值都一起变大或者一起变小?



(a) 原灰度图像　(b) 图像变亮　(c) 图像变暗

图 3-11　使用 brighten 函数使图像变亮或变暗

3.1.3　彩色图像增强

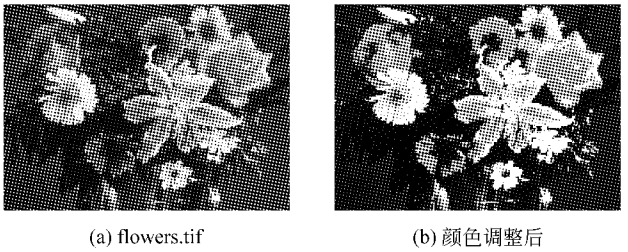
1. RGB 彩色图像增强

【例 3-8】 使用函数 imadjust 对 RGB 彩色图像进行颜色调整。

设计下面程序：

```
RGB1=imread('flowers.tif');
RGB2=imadjust(RGB1,[.2 .3 0; .6 .7 1]);
subplot(1,2,1)
imshow(RGB1)
subplot(1,2,2)
imshow(RGB2)
```

程序运行结果如图 3-12 所示。图像文件 flowers. tif 存储着一幅彩色图像，把该文件读入后使用 imadjust 函数进行调整。调整后的范围是默认的范围[0 0 0; 1 1 1]，也就是把[0.2 0.6]之间的红色映射到[0 1]之间；把[0.3 0.7]之间的绿色映射到[0 1]之间；把[0 1]之间的蓝色映射到[0 1]之间。映射后的结果为图 3-12(b)，从图形的效果上看，红色与黄色成分有所增强，而蓝色成分没有变。



(a) flowers.tif　(b) 颜色调整后

图 3-12　imadjust 函数对 RGB 彩色图像的颜色进行调整(1)

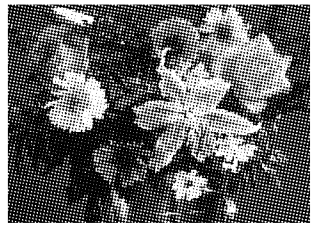
如果把语句

```
RGB2=imadjust(RGB1,[.2 .3 0; .6 .7 1]);
```

修改为：

```
RGB2=imadjust(RGB1,[.2 .3 0; .6 .7 1],[0 0 0; 1 1 0]);
```

那么绘制出图 3-13 所示图像，此时蓝色成分已经全部消失。


(a) flowers.tif


(b) 颜色调整去掉蓝色后

图 3-13 imadjust 函数对 RGB 彩色图像的颜色进行调整(2)

2. HSV 彩色图像增强

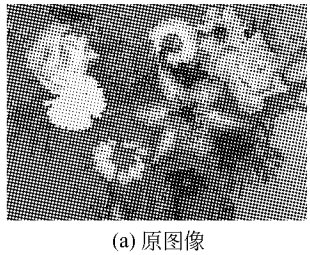
HSV 是另外一种非常重要的彩色图像表示形式,也是由三个矩阵表达彩色信息,分别表达图像的色调、饱和度与颜色值,一般把三个分量矩阵分别记为 H、S、V。

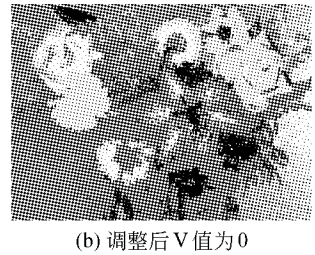
【例 3-9】 使用函数 imadjust 对 HSV 彩色图像进行颜色调整。

设计下面程序:

```
RGB1=imread('flowers.tif');
HSV1=rgb2hsv(RGB1);
HSV2=imadjust(HSV1,[.2 .3 0; .6 .7 1],[0 0 0; 1 1 0]);
HSV3=imadjust(HSV1,[.2 .3 0; .6 .7 1],[0 0 0; 1 0 1]);
HSV4=imadjust(HSV1,[.2 .3 0; .6 .7 1],[0 0 0; 0 1 1]);
subplot(2,2,1); imshow(HSV1)
subplot(2,2,2); imshow(HSV2)
subplot(2,2,3); imshow(HSV3)
subplot(2,2,4); imshow(HSV4)
```

程序运行结果如图 3-14 所示。


(a) 原图像


(b) 调整后V值为0


(c) 调整后S值为0

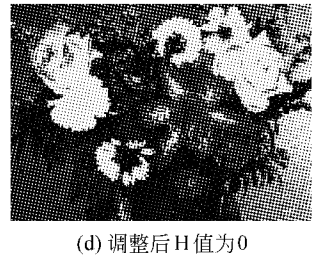

(d) 调整后H值为0

图 3-14 使用函数 imadjust 对 HSV 彩色图像进行颜色调整