实验一 图像处理基础编程

1913416 阎丝雨

一、实验目的

掌握图像读取、图像显示保存、图像灰度化、卷积操作、图像金字塔构建以及傅里叶变换等基础编程算法。

二、实验原理、实验步骤、程序代码、结果显示

（1）图像的读取、显示、保存和灰度化

- 代码如下

```
%图像读取
img=imread("people.bmp");
img_s=imread("scenery.png");

%图像的显示
figure(1)
subplot(1,2,1),imshow(img),title('人像图片');
subplot(1,2,2),imshow(img_s),title('风景图片');

%图像的储存，以人像图为例
imwrite(img,"meinv.jpg")

%%将图像灰度化
img_1=rgb2gray(img);
figure(2),subplot(1,2,1),imshow(img_1),title('灰度图像')

%%将图像灰度化
img_2=rgb2gray(img_s);
figure(2),subplot(1,2,2),imshow(img_2),title('灰度图像')
```

➢ 运行结果：

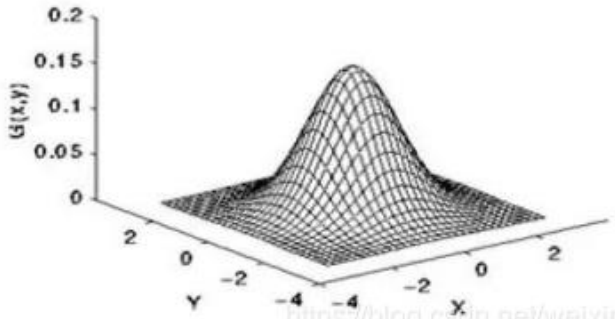
（2）使用不同标准差的高斯模板对图像进行卷积

高斯滤波是一种线性平滑滤波，被广泛应用于图像处理的减噪过程。通俗的讲，高斯滤波就是对整幅图像进行加权平均的过程，每一个像素点的值，都由其本身和邻域内的其他像素值经过加权平均后得到。

高斯滤波的具体操作是：用一个模板（或称卷积、掩模）扫描图像中的每一个像素，用模板确定的邻域内像素的加权平均灰度值去替代模板中心像素点的值。

高斯滤波后图像被平滑的程度取决于标准差，由下图公式可知，标准差越大，图像被平滑的程度就越大。它的输出是临域像素的加权平均，同时离中心越近的像素权重越高。

$$G(x,y)=\frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}}$$

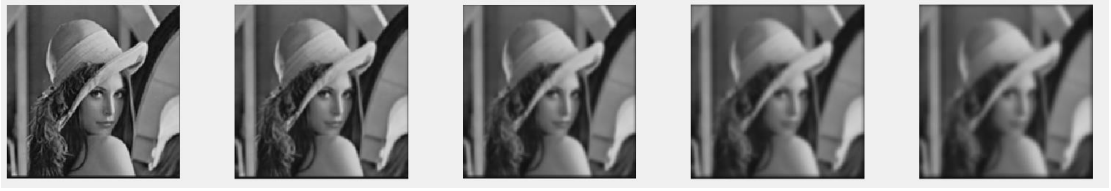


- 代码如下

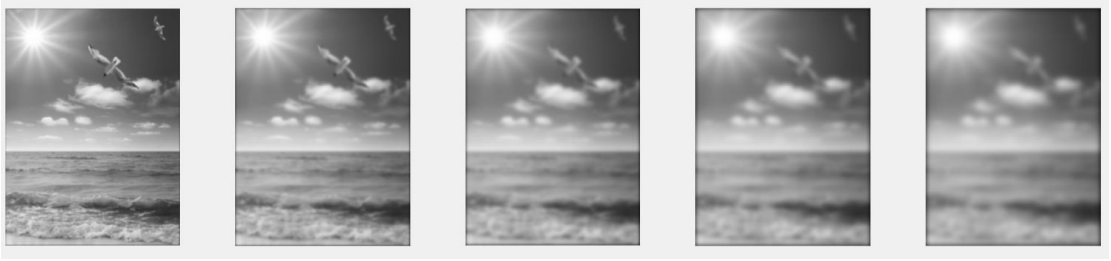
```
%设置高斯模板并卷积(人像图片)
figure(4);
[rows,cols]=size(img_1);
for i=1:5
    sigma=0.5*i;
    gausFilter=fspecial('gaussian',[rows cols],sigma);
    img_3=imfilter(img,gausFilter,'conv');%滤波后的图像
    subplot(1,5,i),imshow(img_3);
end

%设置高斯模板并卷积(风景图片)
figure(5);
[rows_,cols_]=size(img_2);
for i=1:5
    sigma=2*i;
    gausFilter=fspecial('gaussian',[rows_ cols_],sigma);
    img_4=imfilter(img_2,gausFilter,'conv');%滤波后的图像
    subplot(1,5,i),imshow(img_4);
end
```

➢ 运行结果：



（人像图片从左到右的标准差依次上升 0.5，易观察，图片变得越来越平滑）



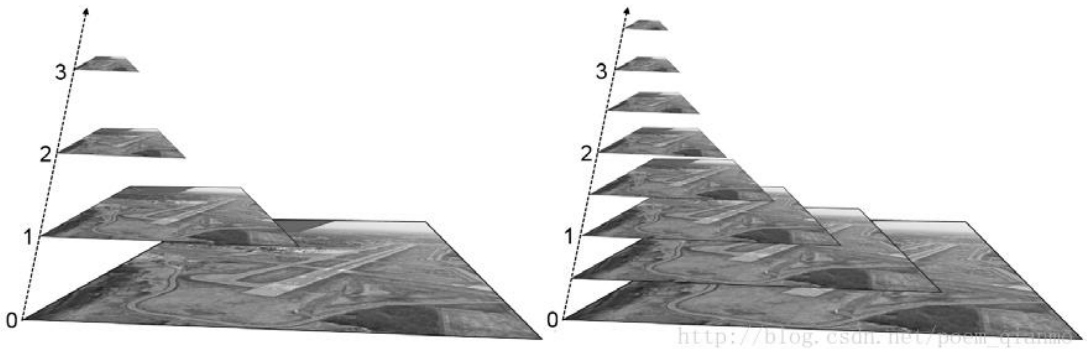
（风景图片从左到右的标准差依次上升 2，易观察，图片变得越来越平滑且平滑的程度远大于人像图像的处理结果）

（3）图像的金字塔表示

图像金字塔是图像中多尺度表达的一种，最主要用于图像的分割，是一种以多分辨率来解释图像的有效但概念简单的结构。

一幅图像的金字塔是一系列以金字塔形状排列的分辨率逐步降低，且来源于同一张原始图的图像集合。其通过梯次向下采样获得，直到达到某个终止条件才停止采样。

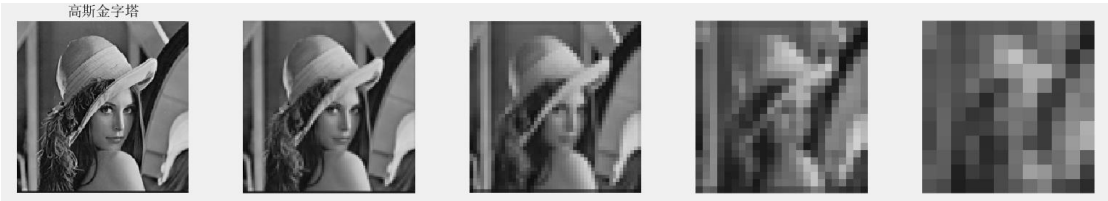
金字塔的底部是待处理图像的高分辨率表示，而顶部是低分辨率的近似。我们将一层一层的图像比喻成金字塔（如下图），层级越高，则图像越小，分辨率越低。

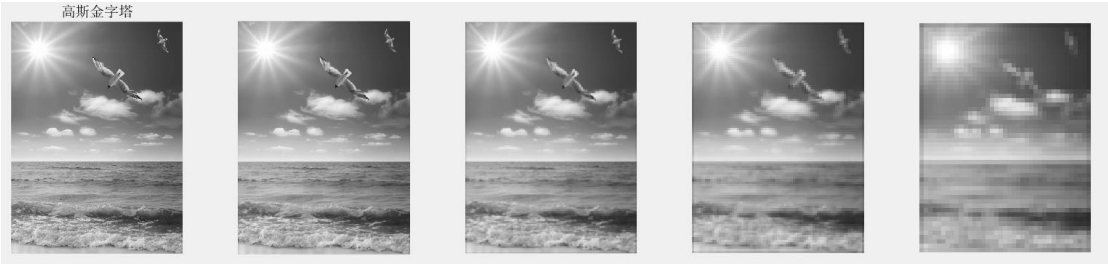


➢ 代码如下

```
%高斯金字塔
[m,n]=size(img_1);
w=fspecial('gaussian',[3 3]);
img2=imresize(imfilter(img_1,w),[m/2 n/2]);
img3=imresize(imfilter(img2,w),[m/4 n/4]);
img4=imresize(imfilter(img3,w),[m/8 n/8]);
img5=imresize(imfilter(img4,w),[m/16 n/16]);
figure(3);
subplot(1,5,1),imshow(img_1),title('高斯金字塔');
subplot(1,5,2),imshow(img2);
subplot(1,5,3),imshow(img3);
subplot(1,5,4),imshow(img4);
subplot(1,5,5),imshow(img5);
```

➢ 运行结果：





（在图片以同样的大小显示时，易观察到图片的分辨率逐步降低）

（4）编写函数去除图片的噪声

➢ 高斯噪声

噪声的概率密度函数服从高斯分布（即正态分布），即某个强度的噪声点个数最多，离这个强度越远噪声点个数越少。

选取均值滤波来处理高斯噪声图片原因是高斯噪声影响了图片的全部像素点，如果只用像素中的中间值来替代所有像素点，会损失掉其他同样受噪声影响的像素点信息，效果不如采用所有像素点平均值的均值滤波。

➢ 椒盐噪声

椒盐噪声也称为脉冲噪声，是图像中经常见到的一种噪声，它是一种随机出现的白点或者黑点，可能是亮的区域有黑色像素或是在暗的区域有白色像素（或是两者皆有）。。椒盐噪声是指两种噪声，盐噪声(高灰度噪声)、胡椒噪声(低灰度噪声)。同时出现时，在图像上呈现为黑白杂点。

选取中值滤波来处理椒盐噪声图片原因是椒盐噪声只影响了图片的部分像素点而不是全部像素点，使用中值滤波方法恰是把数字图像或数字序列中一点的值用该点的一个邻域中各点值的中值代替，让周围像素灰度值的差比较大的像素改取与周围的像素值接近的值，从而可以消除孤立的噪声点。

➢ 程序实现代码如下

```
% 加高斯噪声
Image_noise_gauss = imnoise(img_1,'gaussian'); %加噪
Image_noise_gauss_ = imnoise(img_2,'gaussian'); %加噪
figure(8);
subplot(2,2,1),imshow(Image_noise_gauss),title('高斯噪声图像');
subplot(2,2,2),imshow(Image_noise_gauss_),title('高斯噪声图像');

%均值滤波
img_6=avg_filter(Image_noise_gauss,3 );
img_7=avg_filter(Image_noise_gauss_,3 );
subplot(2,2,3),imshow(img_6),title('均值滤波去噪图像')
subplot(2,2,4),imshow(img_7),title('均值滤波去噪图像')

% 加椒盐噪声
Image_noise_salt = imnoise(img_1,'salt & pepper'); %加噪
Image_noise_salt_ = imnoise(img_2,'salt & pepper'); %加噪
figure(9);
subplot(2,2,1),imshow(Image_noise_salt),title('椒盐噪声图像');
subplot(2,2,2),imshow(Image_noise_salt_),title('椒盐噪声图像');

%中值滤波
img_8=median_filter(Image_noise_salt, 3 );
img_9=median_filter(Image_noise_salt_, 3 )
subplot(2,2,3),imshow(img_8),title('中值滤波去噪图像')
subplot(2,2,4),imshow(img_9),title('中值滤波去噪图像')
```

➢ 均值滤波函数

```
function [d]=avg_filter(x,n)
a(1:n,1:n)=1;    %a即n×n模板,元素全是1
[height, width]=size(x);    %输入图像是hightxwidth的,且hight>n,width>n
x1=double(x);
x2=x1;
for i=1:height-n+1
    for j=1:width-n+1
        c=x1(i:i+(n-1),j:j+(n-1)).*a;
        %取出x1中从(i,j)开始的n行n列元素与模板相乘
        s=sum(sum(c));
        %求c矩阵中各元素之和
        x2(i+(n-1)/2,j+(n-1)/2)=s/(n*n);
        %将与模板运算后的各元素的均值赋给模板中心位置的元素
    end
    d=uint8(x2);
end
```

➢ 中值滤波函数

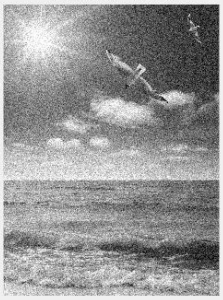
```
function [ img ] = median_filter( image, m )
    n = m;
    [ height, width ] = size(image);
    x1 = double(image);
    x2 = x1;
    for i = 1: height-n+1
        for j = 1:width-n+1
            mb = x1( i:(i+n-1),  j:(j+n-1) );
            mb = mb(:);
            mm = median(mb);
            x2( i+(n-1)/2,  j+(n-1)/2 ) = mm;

        end
    end
    img = uint8(x2);
end
```

➢ 运行结果：

高斯噪声图像

高斯噪声图像



均值滤波去噪图像

均值滤波去噪图像

椒盐噪声图像

椒盐噪声图像

中值滤波去噪图像

中值滤波去噪图像

三、实验分析总结

此次实验实现了一些简单的图像处理操作、例如图像读取与保存、灰度化。并实现了对图像的高斯卷积、金字塔显示、去除噪音等基础操作。