数字图像处理实验报告二

药 3 李宇晖 2013012470

实验目的

了解图像直方图的基本意义和直方图均衡算法

了解空域滤波的概念和算法

实验过程

题目一、实现全局直方图均衡，使用给定图像(pollen1.tif, pollen2.tif, pollen3.tif, pollen4.tif)做实验。请勿使用 matlab 图像处理工具箱函数 histeq.

首先获取目录中所有以'.tif'结尾的文件，并用 for 循环逐一遍历，进行直方图均衡化操作。

实现代码：

```
Files = dir(strcat(root ,'*.tif'));
LengthFile = length(Files);
for loop = 1:LengthFile,
    image = imread(Files(loop).name); %read image
    [m,n,value] = size(image); %get size
    if (value==3),
        image = rgb2gray(image); %change to gray
    end
end
```

对于每一个图像，需要统计各个像素的出现次数，实现代码如下：

```
    Pixel_Count = zeros(1, 256); %save the pixel value and count, note 1:256 in range,
but 0:255 in pixel values
    for i = 1:m,
        for j = 1:n,
            Pixel_Count(image(i,j)+1) = Pixel_Count(image(i,j)+1) + 1;
        end
    end
```

注意图像的像素值是从 0 到 255 的，而 matlab 的向量索引是从 1 到 256 的。

用同样的方法，可以计算出图像的累计像素分布概率，从而得到均衡直方图的映射。实现代码如下：

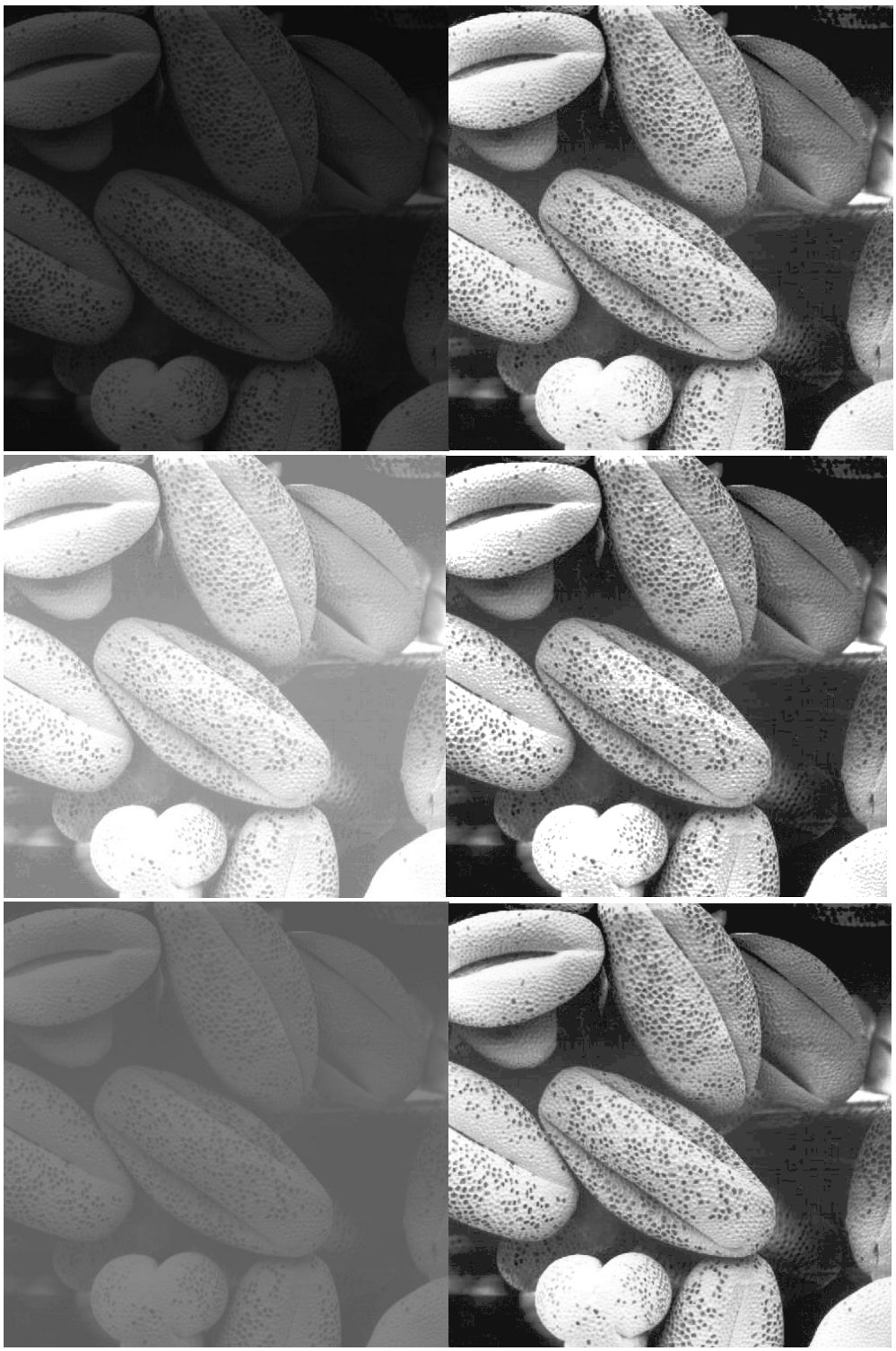
```
    Hist_eq = zeros(1, 256); %save the histogram
    for i = 1: 256,
        if i == 1,
            Hist_eq(i) =    Pixel_Count(i);
        else
            Hist_eq(i) = Pixel_Count(i) + Hist_eq(i-1); %probability
        end
    end
    Hist_eq = uint8(Hist_eq * 255 / (m*n) +0.5); %get inteager and take the floor()
```

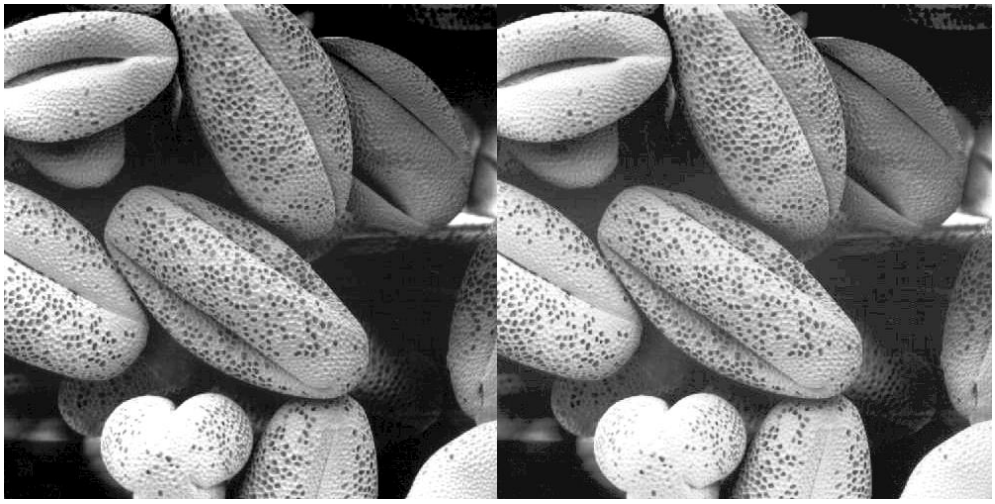
最后创建一个新的图像矩阵，将原图像通过 Hist_eq 映射过去。代码如下：

```
    image_C = image; %save the final image after histeq
    for i = 1:m,
```

```
            for j = 1:n,
                image_C(i,j) = Hist_eq(image(i,j)+1);
            end
        end
        imwrite(image_C,['hist_eq_',Files(loop).name]);
```

实验结果如下（左边为原图，右边为直方图均衡化后的图片）：





可以看出，经过直方图均衡化后，原先的对比度、亮度的差别在映射后几乎消失了。

题目二、对给定人物照片(messi.bmp)的背景做更大程度的虚化（建议尝试多种尺 寸的滤波器）。讨论你所实现算法的不足和可能的改进方法。提示：手工标出前景人物的轮廓；轮廓文件（二值图）需要提交。

首先通过手工交互式的方式勾出前景人物的轮廓，matlab 中 ginput()和 roipoly()函数可以方便地达到目的。代码如下：

```
function [ binary_image ] = Get_FrontGround( image )
    close all;
    imshow(image);
    [x,y] = ginput;
    binary_image = roipoly(image,x,y);
    imwrite(binary_image,'binary.bmp');
    binary_image = 1 - binary_image;
    imwrite(binary_image,'without_front.bmp');
end
```

得到二值图如下：

然后再通过空域滤波对背景进行虚化处理。由于本题不限制 matlab 工具箱函数的使用，所以调用函数实现。若不调用，也可自己编写函数，中值滤波的代码如下：

```
function [ final_image ] = Median_Filter( image, filter_size )
```

```
    [height, width] = size(image);
    image = double(image);
    final_image = image;
    for i = 1:height - filter_size + 1,
        for j = 1:width - filter_size + 1,
            square = image(i:(i + filter_size - 1), j:(j + filter_size - 1));
            median_square = median(median(square));   %use median value to replace
image
            final_image(i + filter_size - 1, j + filter_size -1) = median_square;
        end
    end
    final_image = uint8(final_image);
end
```

在 main 函数中，将原图点乘 1-二值图即可得到前景，点乘二值图即可得到背景。实现代码如下：

```
messi_image = imread('messi.bmp'); %read image
[m,n,c] = size(messi_image);
if c==3,
    messi_image = rgb2gray(messi_image); %promise changing to gray
end
binary_image = Get_FrontGround(messi_image); %get binary front image
binary_image = imread('without_front.bmp')/255;
front_image = uint8(messi_image) .* uint8(1-binary_image);
image_process = uint8(messi_image) .* uint8(binary_image); %get image to process
```

而后，用循环的方式分别不同大小的滤波器对原图进行滤波。这里实验采用三种常见的滤波器：均值滤波、中值滤波和线性平滑滤波。实现代码如下：

```
Filter_Range = [6, 20, 30, 50];
for i = 1:4,
    filter_size = Filter_Range(i);   %filter size 6*6, 20*20, 30*30, 50*50
    median_image = medfilt2(image_process,[filter_size,filter_size]); %median filter
    imwrite(median_image+front_image,['median',num2str(i),'.bmp']);
    linear_image = filter2(fspecial('average',filter_size),image_process)/255; %linear filter
    imwrite(linear_image+double(front_image)/255,['linear',num2str(i),'.bmp']);
    filter_size = 3*i; % filter size is 3, 6, 9, 12
    mean_image = imfilter(image_process,i);%mean filter
    imwrite(mean_image+front_image,['mean',num2str(i),'.bmp']);
end
```

在实际操作过程中，由于滤波器的改变不大不能看出差异，还在控制台重新尝试了几种滤波器的大小。实验结果如下：

线性平滑滤波结果

滤波器大小 6*6

滤波器大小 20*20

滤波器大小 30*30

滤波器大小 50*50

均值滤波结果

滤波器大小 3*3

滤波器大小 6*6

滤波器大小 9*9

滤波器大小 12*12

中值滤波结果

滤波器大小 6*6

滤波器大小 20*20

滤波器大小 30*30

滤波器大小 50*50

可以看出，不论哪种滤波方法，当滤波器大小越大时，图像就会越模糊。其中，均值滤波在图像的对比度很强时，会偏向于白色。实际上，还可以尝试其他的滤波方法。同时，由于人工勾勒的前景并不足够理想，会导致整体的画面很突兀。可以尝试使用差分等算法提取前景图像，再进行模糊。